

BBBBBBBBBBBBBBB AAAAAAAA SSSSSSSSSSSS RRRRRRRRRRRRR TTTTTTTTTTTTTTTT LLL
BBBBBBBBBBBBBBB AAAAAAAA SSSSSSSSSSSS RRRRRRRRRRRRR TTTTTTTTTTTTTTTT LLL
BBBBBBBBBBBBBBB AAAAAAAA SSSSSSSSSSSS RRRRRRRRRRRRR TTTTTTTTTTTTTTTT LLL

BBB BBB AAA AAA SSS

BBBBBBBBBBBBBBB AAA AAA SSSSSSSSSS
BBBBBBBBBBBBBBB AAA AAA SSSSSSSSSS
BBBBBBBBBBBBBBB AAA AAA SSSSSSSSSS

BBB BBB AAAAAAAAAAAAAA SSS
BBB BBB AAAAAAAAAAAAAA SSS
BBB BBB AAAAAAAAAAAAAA SSS
BBB BBB AAA AAA SSS
BBB BBB AAA AAA SSS
BBB BBB AAA AAA SSS

BBBBBBBBBBBBBBB AAA AAA SSSSSSSSSSSS
BBBBBBBBBBBBBBB AAA AAA SSSSSSSSSSSS
BBBBBBBBBBBBBBB AAA AAA SSSSSSSSSSSS

FILEID**BASRECPRO

F 4

BBBBBBBBBB	AAAAAA	SSSSSSSS	RRRRRRRR	EEEEEEEEE	CCCCCCCC	PPPPPPPP	RRRRRRRR	000000
BBBBBBBBBB	AAAAAA	SSSSSSSS	RRRRRRRR	EEEEEEEEE	CCCCCCCC	PPPPPPPP	RRRRRRRR	000000
BB BB	AA AA	SS	RR RR	EE	CC	PP PP	RR RR	00 00
BB BB	AA AA	SS	RR RR	EE	CC	PP PP	RR RR	00 00
BB BB	AA AA	SS	RR RR	EE	CC	PP PP	RR RR	00 00
BB BB	AA AA	SS	RR RR	EE	CC	PP PP	RR RR	00 00
BBBBBBBBBB	AA AA	SSSSSS	RRRRRRRR	EEEEEEEEE	CC	PPPPPPPP	RRRRRRRR	00 00
BBBBBBBBBB	AA AA	SSSSSS	RRRRRRRR	EEEEEEEEE	CC	PPPPPPPP	RRRRRRRR	00 00
BB BB	AAAAAAA	SS	RR RR	EE	CC	PP	RR RR	00 00
BB BB	AAAAAAA	SS	RR RR	EE	CC	PP	RR RR	00 00
BB BB	AA AA	SS	RR RR	EE	CC	PP	RR RR	00 00
BB BB	AA AA	SS	RR RR	EE	CC	PP	RR RR	00 00
BBBBBBBBBB	AA AA	SSSSSSSS	RR RR	EEEEEEEEE	CCCCCCCC	PP	RR RR	000000
BBBBBBBBBB	AA AA	SSSSSSSS	RR RR	EEEEEEEEE	CCCCCCCC	PP	RR RR	000000

LL		SSSSSSSS
LL		SSSSSSSS
LL		SS
LL		SS
LL		SS
LL		SSSSSS
LL		SSSSSS
LL		SS
LL		SS
LL		SS
LLLLLLLLL		SSSSSSSS
LLLLLLLLL		SSSSSSSS

```
1 0001 0 MODULE BASS$REC_PROC (
2 0002 0           IDENT = '1-095'
3 0003 0           ) =
4 0004 1 BEGIN
5
6 0006 1 ****
7 0007 1 *
8 0008 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
9 0009 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
10 0010 1 * ALL RIGHTS RESERVED.
11 0011 1 *
12 0012 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
13 0013 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
14 0014 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
15 0015 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
16 0016 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
17 0017 1 * TRANSFERRED.
18 0018 1 *
19 0019 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
20 0020 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
21 0021 1 * CORPORATION.
22 0022 1 *
23 0023 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
24 0024 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
25 0025 1 *
26 0026 1 *
27 0027 1 ****
28 0028 1 ++
29 0029 1 FACILITY: BASIC Support Library - not user callable
30 0030 1 ABSTRACT:
31 0031 1
32 0032 1 This module implements the record processing level of
33 0033 1 abstraction which is the 3rd level and is called only from
34 0034 1 the user data formatter level (2nd level) when the user
35 0035 1 portion of a record buffer is full (WRITE) or empty
36 0036 1 (READ). This module adds any per record formatting (as
37 0037 1 distinguished from per I/O statement or per I/O list element
38 0038 1 formatting) and then calls RMS ($PUT or $GET). RMS errors
39 0039 1 are converted to BASIC errors and are signaled.
40 0040 1
41 0041 1 ENVIRONMENT: User access mode; AST level or not.
42 0042 1
43 0043 1 AUTHOR: Donald G. Petersen; CREATION DATE: 16-Mar-78
44 0044 1
45 0045 1 MODIFIED BY:
46 0046 1
47 0047 1 0-61 - Add ATTEMPT TO READ NON-EXISTANT RECORD error to seq. reads.
48 0048 1 JMT 02-Jan-78
49 0049 1 Donald G. Petersen, 16-Mar-78 : VERSION 1-01
50 0050 1
51 0051 1 1-01 - original DGP
52 0052 1 1-02 - Change to JSB linkages. DGF 14-Nov-78
53 0053 1 1-004 - Update copyright notice and add device names to REQUIRE
54 0054 1 files. JBS 29-NOV-78
55 0055 1 1-005 - Add BASS$RECOUNT to support the Basic RECOUNT function. DGP
56 0056 1 03-Dec-78
57 0057 1
```

58 0058 1 | 1-006 - Dot problem in BASSRECOUNT. DGP 04-Dec-78
59 0059 1 | 1-007 - Add fudge factor to RECOUNT for the line terminator if input is
60 0060 1 | from a terminal. DGP 04-Dec-78
61 0061 1 | 1-008 - Change REQUIRE file names from FOR... to OTS... JBS 07-DEC-78
62 0062 1 | 1-009 - Call BASS\$SIGNAL_IO for RMS errors. JBS 18-DEC-78
63 0063 1 | 1-010 - Add new routines for READ. DGP 19-Dec-78
64 0064 1 | 1-011 - Change references to ISBSA_BUF_PTR, BUF_BEG, BUF_END to LUB.
65 0065 1 | DGP 05-Jan-78
66 0066 1 | 1-012 - Change to CR format for terminal data files. DGP 11-Jan-79
67 0067 1 | 1-013 - Make a few changes for recursive I/O. DGP 15-Jan-79
68 0068 1 | 1-014 - Remove signalling for *Z. Moved to UDF level for INPUT LINE hand-
69 0069 1 | ling. DGP 15-Jan-79
70 0070 1 | 1-015 - Change stack frame prefix to BSF\$. JBS 08-FEB-1979
71 0071 1 | 1-016 - Add BASS\$REC_GSE (Basic GET sequential). DGP 19-Feb-79
72 0072 1 | 1-017 - Add BASS\$REC_PSE (Basic PUT sequential). DGP 20-Feb-79
73 0073 1 | 1-018 - set RABSL_RBF in BASS\$REC_PSE. DGP 20-Feb-79
74 0074 1 | 1-019 - Set RABSL_RBF in BASS\$GSE. DGP 21-Feb-79
75 0075 1 | 1-020 - Null fill buffer for GET. DGP 27-Feb-79
76 0076 1 | 1-021 - Add REC routines for FIND, DELETE, UPDATE, RESTORE, SCRATCH. DGP
77 0077 1 | 27-Feb-79
78 0078 1 | 1-022 - Add BASIOERR.REQ for error handling. DGP 28-Feb-79
79 0079 1 | 1-023 - Add BASUNLOCK and BASFREE. DGP 28-Feb-79
80 0080 1 | 1-024 - Set RABSL_RBF in BASS\$REC_UPD. DGP 01-Mar-79
81 0081 1 | 1-025 - Add REC_PRE, REC_GRE, REC_FRE. DGP 02-Mar-79
82 0082 1 | 1-026 - More work on relative I/O. DGP 05-Mar-79
83 0083 1 | 1-027 - Update pointer for READ in Basic Major Frame in RMF9. DGP 06-Mar-79
84 0084 1 | 1-028 - Add support for Basic "foreign buffers". DGP 27-Mar-79
85 0085 1 | 1-029 - Point all GETs and PUTs off to GET_ERROR or PUT_ERROR. DGP 02-Apr-79
86 0086 1 | 1-030 - Add more routines to support ISAM. DGP 03-Apr-79
87 0087 1 | 1-031 - Fix PUT sequential to support ISAM. DGP 04-Apr-79
88 0088 1 | 1-032 - Put in indexed I/O stuff. 06-Apr-79
89 0089 1 | 1-033 - Bug fixes in indexed. 10-Apr-79 DGP
90 0090 1 | 1-034 - Implement the WAIT statement, using LUB\$L_WAIT_TIME.
91 0091 1 | JBS 10-APR-1979
92 0092 1 | 1-035 - Implement the ECHO and NOECHO functions, using LUB\$V_NOECHO.
93 0093 1 | JBS 17-APR-1979
94 0094 1 | 1-036 - Add code to handle single-character input from GET
95 0095 1 | SEQUENTIAL. JBS 17-APR-1979
96 0096 1 | 1-037 - Implement the CTRLO and RCTRLO functions, using LUB\$V_CCO.
97 0097 1 | JBS 19-APR-1979
98 0098 1 | 1-038 - Implement the Cancel Typeahead function, using LUB\$V_PTA.
99 0099 1 | JBS 01-MAY-1979
100 0100 1 | 1-039 - Add Basic PRINT USING support. DGP 15-May-79
101 0101 1 | 1-040 - Change BIND to GLOBAL BIND ROUTINE in PRINT USING support.
102 0102 1 | JBS 16-MAY-1979
103 0103 1 | 1-041 - Add BASSRECOU_INIT. JBS 04-JUN-1979
104 0104 1 | 1-042 - Add REC level for MAT INPUT. DGP 05-Jun-79
105 0105 1 | 1-043 - Clean up a lot and put real code into Matrix Input routines. DGP
106 0106 1 | 14-Jun-79
107 0107 1 | 1-044 - Make REC_MIN1 look for continuation character. DGP 20-Jun-79
108 0108 1 | 1-045 - Terminal devices use PRN format for output. DGP 10-Jul-79
109 0109 1 | 1-046 - Add BASS\$NUM INIT, BASS\$NUM2 INIT, BASSMAT_LINPUT, BASSMAT_READ,
110 0110 1 | BASS\$NUM, BASSNUM. DGP 13-Jul-79
111 0111 1 | 1-047 - Change ISBSL MAJ_F PTR to ISBSA MAJ_F PTR. JBS 24-JUL-1979
112 0112 1 | 1-048 - Signal if READ with no DATA. DGP 07-Aug-79
113 0113 1 | 1-049 - Debug MAT I/O. DGP 07-Aug-79
114 0114 1 | 1-050 - STOP a few errors that are being SIGNALLED. DGP 05-Sep-79

115 0115 1 1-051 - FREE and UNLOCK are noops if no record locked. DGP 06-Sep-79
116 0116 1 1-052 - Move NUM_INIT and NUM2_INIT to BASSMAT IO, and move BASS\$BLNK_LINE
117 0117 1 from BASSMAT IO to here. DGP 06-Sep-79
118 0118 1 1-053 - Load LUBSA_RBUF_ADR for GET and PUT for Locate mode (RMS). DGP
119 0119 1 13-Sep-79
120 0120 1 1-054 - Clear the prompt buffer in GET_ERROR. DGP 17-Sep-79
121 0121 1 1-055 - Fix BASS\$BLNK_LINE. DGP 04-Oct-79
122 0122 1 1-056 - Add MAT READ. DGP 11-Oct-79
123 0123 1 1-057 - Add a REC9 routine for MAT PRINT. DGP 12-Oct-79
124 0124 1 1-058 - Add BASS\$REC_MLI1. DGP 12-Oct-79
125 0125 1 1-059 - Fix BASS\$REC_WSL1 to leave the cursor alone. DGP 02-Nov-79
126 0126 1 1-060 - Allow BASS\$REC_WSL1 to accept an argument. DGP 06-Nov-79
127 0127 1 1-061 - GET will only null fill the buffer, if necessary, after a GET. DGP
128 0128 1 12-Nov-79
129 0129 1 1-062 - BASS\$REC_MPR1 needs an LF if no format char. DGP 13-Nov-79
130 0130 1 1-063 - Use LUBSA UBF to simplify foreign buffer code. JBS 13-NOV-1979
131 0131 1 1-064 - BASS\$REC_MIN1 should not differentiate between terminal & non-
132 0132 1 terminal devices. DGP 14-Nov-79
133 0133 1 1-065 - GET relative not null filling the buffer properly. DGP 29-Nov-79
134 0134 1 1-066 - Null fill the buffer before restoring foreign buffer pointers for
135 0135 1 GET. DGP 18-Dec-79
136 0136 1 1-067 - RMS does not return a terminator in the STV field for files. DGP
137 0137 1 03-Jan-80
138 0138 1 1-068 - REC_WSL9 should only write a record if the output buffer has some-
139 0139 1 thing in it to write. DGP 03-Jan-80
140 0140 1 1-069 - Addition to 1-068. Should also write a record if there was no element
141 0141 1 transmitter. DGP 04-Jan-80
142 0142 1 1-070 - Unconditionally write a CR in WSL1. DGP 14-Jan-80
143 0143 1 1-071 - Restore "foreign buffers" properly and set RECOUNT in GET Indexed
144 0144 1 and Relative. DGP 12-Feb-80
145 0145 1 1-072 - Adjust the Global RECOUNT to include the length of an escape sequence.
146 0146 1 DGP 22-Feb-80
147 0147 1 1-073 - A previous edit to fix a problem with foreign buffers reintroduced
148 0148 1 a problem with only null padding the buffer after a successful GET.
149 0149 1 DGP 26-Feb-80
150 0150 1 1-074 - REC_WSL9 should set VFC2 to BASS\$NULL (no carriage control) if there
151 0151 1 is a format character. DGP 26-Feb-80
152 0152 1 1-075 - Update the cursor position for INPUT if terminated by an escape.
153 0153 1 DGP 27-Feb-80
154 0154 1 1-076 - When calculating CPOS (current cursor position) following an INPUT
155 0155 1 statement, take the prompt string into account.
156 0156 1 1-077 - REC_RSL1 is not updating the cursor position correctly. DGP 04-Mar-80
157 0157 1 1-078 - REC_WSL9 should set the 'pre' carriage control for the next record
158 0158 1 to LF if there is no format character 'cuz of recursive I/O. DGP
159 0159 1 07-Mar-80
160 0160 1 1-079 - Rationalize the CCO and PTA bits. CCO is now copied from LUB to RAB
161 0161 1 when initializing for output, and PTA when initializing for input.
162 0162 1 JBS 31-MAR-1980
163 0163 1 1-080 - Clear the dirty bit CCB [LUB\$V_OUTBUF_DR] in PUT_ERROR so BASSCLOSE
164 0164 1 when invoked by the unwind won't get confused and do a PUT.
165 0165 1 FM 11-SEP-80
166 0166 1 1-081 - Tack on the terminator(s) to the buffer when a GET is done on a
167 0167 1 terminal device file in BASS\$REC_GSE.
168 0168 1 1-082 - Add/transfer BASS\$WAIT to this module, now wait routines are part of
169 0169 1 the sharable image. The routines added are BASS\$WAIT,
170 0170 1 BASS\$READ_WAIT. We had to make WAIT routines part of the sharable
171 0171 1 image because WAIT was requested to become a GLOBAL, and routines in

172 0172 1 this module had to read it.
173 0173 1
174 0174 1 1-083- Only if LUB\$B_RAT indicates CR format tack on the CRLF. FM 9-feb-81
175 0175 1 1-084- Cursor position not updated correctly if INPUT was
176 0176 1 terminated by an escape - code should check if previous
177 0177 1 PRINT was terminated by a semicolon or comma. PLL 5-7-81
178 0178 1 1-085- The purge typeahead function is no longer setting the PTA bit in
179 0179 1 the LUB, so BASS\$REC_RSL0, BASS\$REC_MIN0, and BASS\$REC_GSE don't need
180 0180 1 to check it anymore. PLL 6-Aug-81
181 0181 1 1-086 - Add support for RFA access and manual record locking. PLL 1-Jun-82
182 0182 1 1-087 - BASS\$REC_GIN and BASS\$REC_FIN should check for a decimal key
183 0183 1 when setting the key size in the RAB. PLL 6-Jul-1982
184 0184 1 1-088 - Add support for ANSI INPUT. If not enough data is supplied, the
185 0185 1 entire INPUT must be restarted. PLL 29-Jul-1982
186 0186 1 1-089 - Fix CTRL/O rationalization. Unconditionally copy whatever its
187 0187 1 state is in the LUB into the RAB, and clear it in the LUB.
MDL and JBS 10-Aug-1982
188 0188 1 1-090 - Set buffer pointer (in WSL9) from buffer beginning pointer rather
189 0189 1 than from RBUF ADR. This fixes the problem of pointing to the
190 0190 1 wrong buffer when the user enters a CTRL/C while a line is being
191 0191 1 written, and his control-c routine writes a line also.
MDL and PLL 19-Aug-1982
192 0192 1
193 0193 1 1-091 - In BASS\$REC_RSL1, BASS\$SIGNAL_10 the too little data error instead
194 0194 1 of calling BASS\$STOP_10. PLL 27-Sep-1982
195 0195 1 1-092 - In BASS\$REC_RSL0, the contents of the print buffer should be \$PUT
196 0196 1 before the \$GET is done, if this is a non-terminal device. This
197 0197 1 is a requested behavior change that will cause input prompts to
198 0198 1 appear in batch log files. RMS is making a change concurrently
199 0199 1 that will cause the actual input provided to appear in the batch
200 0200 1 log as well, thus making a batch log an exact duplicate of how
201 0201 1 it would appear if run interactively. MDL 26-Jul-1983
202 0202 1 1-093 - check for RMS\$_CONTROL_C completion status; call new CTRL/C signaller
203 0203 1 if this status is returned. this change is coordinated with rev.
204 0204 1 2-003 of BAS\$CTRLC. MDL 12-Mar-1984
205 0205 1 1-094 - for the special case of channel 0, edit 1-092 should reach thru the
206 0206 1 buddy ptr and use the output side of channel 0 to write out the
207 0207 1 prompt string. MDL 22-Mar-1984
208 0208 1 1-095 - routines that set and reset record options should reset them BEFORE
209 0209 1 calling error routines, so that subsequent I/O on the channel will
210 0210 1 work properly (if the user handles the error). MDL 23-Mar-1984
211 0211 1 --
212 0212 1
213 0213 1 !<BLF/PAGE>

```

215 0214 1 | SWITCHES:
216 0215 1 | .SWITCHES:
217 0216 1 | .SWITCHES ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
218 0217 1 |
219 0218 1 | LINKAGES
220 0219 1 |
221 0220 1 |
222 0221 1 | REQUIRE 'RTLIN:OTSLNK';                                ! define all linkages
223 0222 1 |
224 0223 1 |
225 0224 1 | TABLE OF CONTENTS:
226 0225 1 |
227 0226 1 |
228 0227 1 |
229 0228 1 |
230 0229 1 |
231 0230 1 | FORWARD ROUTINE
232 0231 1 | BASSWAIT : NOVALUE,
233 0232 1 | BAS$READ_WAIT,
234 0233 1 | BAS$RECOUNT
235 0234 1 | BAS$RECOUN INIT : NOVALUE,
236 0235 1 | BAS$BLNK_LINE : CALL_CCB NOVALUE,
237 0236 1 | ! write sequential list-directed
238 0237 1 | BAS$REC_WSL0 : JSB_REC0 NOVALUE,
239 0238 1 | BAS$REC_WSL1 : JSB_REC WSL1 NOVALUE,
240 0239 1 | BAS$REC_WSL9 : JSB_REC9 NOVALUE,
241 0240 1 | ! Mat Input
242 0241 1 | BAS$REC_MLI1 : JSB_REC1,
243 0242 1 | ! Mat Read
244 0243 1 | BAS$REC_MRE1 : JSB_REC1,
245 0244 1 | ! Mat Print
246 0245 1 | BAS$REC_MPR1 : JSB_REC1 NOVALUE,
247 0246 1 | BAS$REC_MPR9 : JSB_REC9 NOVALUE,
248 0247 1 | ! read sequential list-directed
249 0248 1 | BAS$REC_RSL0 : JSB_REC0 NOVALUE,
250 0249 1 | BAS$REC_RSL1 : JSB_REC1,
251 0250 1 | BAS$REC_RSL9 : JSB_REC9 NOVALUE,
252 0251 1 | ! MAT INPUT
253 0252 1 | BAS$REC_MIN0 : JSB_REC0 NOVALUE,
254 0253 1 | BAS$REC_MIN1 : JSB_REC1,
255 0254 1 | BAS$REC_MIN9 : JSB_REC9 NOVALUE,
256 0255 1 | ! read memory list-directed
257 0256 1 | BAS$REC_RMF0 : JSB_REC0 NOVALUE,
258 0257 1 | BAS$REC_RMF1 : JSB_REC1 NOVALUE,
259 0258 1 | BAS$REC_RMF9 : JSB_REC9 NOVALUE;
260 0259 1 |
261 0260 1 | GLOBAL BIND
262 0261 1 | ROUTINE
263 0262 1 | ! write formatted
264 0263 1 | BAS$REC_WF0 = BAS$REC_WSL0,
265 0264 1 | BAS$REC_WF1 = BAS$REC_WSL1,
266 0265 1 | BAS$REC_WF9 = BAS$REC_WSL9;
267 0266 1 |
268 0267 1 | FORWARD ROUTINE
269 0268 1 | ! record operations
270 0269 1 | BAS$REC_GSE : JSB_DO_READ NOVALUE,          ! GET sequential
271 0270 1 | BAS$REC_PSE : JSB_PUT NOVALUE,             ! PUT sequential

```

```

272 0699 1 BASS$REC_FSE : JSB REC2 NOVALUE,
273 0700 1 BASS$REC_FRFA: JSB REC2 NOVALUE,
274 0701 1 BASS$REC_DSE : JSB RECO NOVALUE,
275 0702 1 BASS$REC_UPD : JSB DO WRITE NOVALUE,
276 0703 1 BASS$REC_RSE : JSB RECO NOVALUE,
277 0704 1 BASS$REC_SSE : JSB RECO NOVALUE,
278 0705 1 BASS$REC_PRE : JSB PUT NOVALUE,
279 0706 1 BASS$REC_GRE : JSB DO READ NOVALUE,
280 0707 1 BASS$REC_GRFA: JSB DO READ NOVALUE,
281 0708 1 BASS$REC_FRE : JSB RECO NOVALUE,
282 0709 1 BASS$REC_UNL : JSB RECO NOVALUE,
283 0710 1 BASS$REC_FEE : JSB RECO NOVALUE,
284 0711 1 BASS$REC_GIN : JSB REC_IND1 NOVALUE,
285 0712 1 BASS$REC_FIN : JSB REC_IND1 NOVALUE,
286 0713 1 BASS$REC_RIN : JSB REC_IND NOVALUE,
287 0714 1 PUT_ERROR : CALL_CCB NOVALUE,
288 0715 1 GET_ERROR : CALL_CCB NOVALUE;
289 0716 1
290 0717 1 ! INCLUDE FILES:
291 0718 1 ! INCLUDE FILES:
292 0719 1
293 0720 1
294 0721 1 REQUIRE 'RTLIN:BASIOERR'; ! I/O error codes
295 0774 1
296 0775 1 REQUIRE 'RTLIN:BASFRAME'; ! Basic frame offsets
297 0978 1
298 0979 1 REQUIRE 'RTLML:OTSISB'; ! I/O statement block (ISB) offsets
299 1147 1
300 1148 1 REQUIRE 'RTLML:OTSLUB'; ! Logical unit block (LUB) offsets
301 1288 1
302 1289 1 REQUIRE 'RTLIN:OTSMAC'; ! Macros
303 1483 1
304 1484 1 REQUIRE 'RTLIN:RTLPSECT'; ! Define DECLARE_PSECTS macro
305 1579 1
306 1580 1 REQUIRE 'RTLML:BASPAR'; ! BASIC inter-module parameters
307 1602 1
308 1603 1 LIBRARY 'RTLSTARLE'; ! STARLET Library for macros and symbols
309 1604 1
310 1605 1
311 1606 1 ! MACROS:
312 1607 1
313 1608 1 ! MACROS:
314 1609 1 ! MACROS:
315 1610 1 ! EQUATED SYMBOLS:
316 1611 1
317 1612 1
318 1613 1 ! LITERAL
319 1614 1 ! K_MAT_CONT_CHAR = XX'26',
320 1615 1
321 1616 1 ! K_STOP = 0,
322 1617 1 ! K_SIGNAL = 1;
323 1618 1
324 1619 1
325 1620 1 ! PSECT DECLARATIONS:
326 1621 1
327 1622 1 ! DECLARE_PSECTS (BAS); ! declare PSECTS for BASS facility
328 1623 1

```

```
329      1624 1 ! OWN STORAGE:  
330      1625 1  
331      1626 1 OWN  
332      1627 1 RECOUNT : INITIAL (0),  
333      1628 1 WAIT   : WORD INITIAL (0);  
334      1629 1  
335      1630 1 ! EXTERNAL REFERENCES:  
336      1631 1  
337      1632 1  
338      1633 1  
339      1634 1 EXTERNAL ROUTINE  
340      1635 1 BAS$$SIGNAL : NOVALUE,  
341      1636 1 BAS$$STOP IO : NOVALUE,  
342      1637 1 BAS$$SIGNAL_IO : NOVALUE,  
343      1638 1 BAS$$SIGNAL_CTRLC : NOVALUE;  
344      1639 1  
345      1640 1  
346      1641 1  
347      1642 1 EXTERNAL LITERAL  
348      1643 1 BAS$K_OUTOF DAT : UNSIGNED (8),  
349      1644 1 BAS$K_ENDFILEDEV : UNSIGNED (8),  
350      1645 1 BAS$K_NOTENO DAT : UNSIGNED (8),  
351      1646 1 BAS$K_TOOLITDAT : UNSIGNED (8),  
352      1647 1 BAS$K_RECFILETOO : UNSIGNED (8);  
353      1648 1  
354      1649 1 !  
355      1650 1
```

| Signal a BASIC error
| Signal fatal Basic error
| Signal a BASIC I/O error
| Signal CTRL/C

| out of data (READ)
| end of file on device
| not enough data
| ANSI for above
| record in file too long

```

357 1651 1 GLOBAL ROUTINE BASSWAIT (
358 1652 1 TIME
359 1653 1 ) : NOVALUE =
360 1654 1
361 1655 1 ++
362 1656 1 FUNCTIONAL DESCRIPTION:
363 1657 1
364 1658 1 Limits the time any input I/O statement ( INPUT, INPUT LINE, LINPUT,
365 1659 1 MAT 'all above', GET ) to any terminal will wait. If the user does not
366 1660 1 reply before the indicated number of seconds an error trap ( which the
367 1661 1 user can intercept ) will be taken. WAIT is a module level OWN in this
368 1662 1 module.
369 1663 1
370 1664 1
371 1665 1 FORMAL PARAMETERS:
372 1666 1
373 1667 1 TIME.rl.v Number of seconds to wait, max.
374 1668 1
375 1669 1 IMPLICIT INPUTS:
376 1670 1
377 1671 1 The module level OWN WAIT
378 1672 1
379 1673 1 IMPLICIT OUTPUTS:
380 1674 1
381 1675 1 Writes to the module level OWN WAIT the number of seconds given
382 1676 1
383 1677 1 ROUTINE VALUE:
384 1678 1
385 1679 1 None
386 1680 1
387 1681 1 SIDE EFFECTS:
388 1682 1
389 1683 1 None
390 1684 1
391 1685 1 --
392 1686 1
393 1687 2 BEGIN
394 1688 2 +
395 1689 2 If the WAIT time is unreasonable then force it to the acceptable range.
396 1690 2 This is until the correct error message is cooked up for this error.
397 1691 2 WAIT is a module level OWN.
398 1692 2 -
399 1693 2 WAIT = MIN ( ABS(.TIME) , 255 );
400 1694 2 RETURN;
401 1695 2 END;

```

.TITLE BASS\$REC_PROC
.IDENT \1-095\

.PSECT _BASSDATA, NOEXE, PIC.2

00000000 0000 RECOUNT:.LONG 0
0000 0004 WAIT:.WORD 0

.EXTRN BASS\$SIGNAL, BASS\$STOP_10
.EXTRN BASS\$SIGNAL_10, BASS\$SIGNAL_CTRLC

.EXTRN BASSK_OUTOF DAT
.EXTRN BASSK-ENDFI[DEV
.EXTRN BASSK_NOTENODAT
.EXTRN BASSK_TOOLITDAT
.EXTRN BASSK_RECFILETOO
.PSECT _BASS\$CODE,NOWRT, SHR, PIC,2

50 04 0000 00000
000000FF 8F 03 18 00006
00000000' 50 50 CE 00008
EF FF 50 D1 00008 15:
04 15 00012
8F 9A 00014
50 B0 00018 25:
04 0001F

.ENTRY BASSWAIT, Save nothing
MOV_L TIME, R0
BGEQ 1\$
MNEGL R0, R0
CMPL R0, #255
BLEQ 2\$
MOVZBL #255, R0
MOVW R0, WAIT
RET

1651
1693

1695

; Routine Size: 32 bytes, Routine Base: _BASS\$CODE + 0000

; 402 1696 1

```

: 404      1697 1 ROUTINE BASSREAD_WAIT           !Read the module level OWN WAIT
: 405      1698 1
: 406      1699 1
: 407      1700 1 ++
: 408      1701 1 FUNCTIONAL DESCRIPTION:
: 409      1702 1
: 410      1703 1     Read the module level OWN WAIT and return it. The value of this
: 411      1704 1     function is the current contents of wait. All routines that need
: 412      1705 1     this value must call this routine.
: 413      1706 1
: 414      1707 1
: 415      1708 1
: 416      1709 1
: 417      1710 1
: 418      1711 1
: 419      1712 1
: 420      1713 1
: 421      1714 1     Reads the module level OWN WAIT
: 422      1715 1
: 423      1716 1
: 424      1717 1
: 425      1718 1
: 426      1719 1
: 427      1720 1
: 428      1721 1
: 429      1722 1
: 430      1723 1
: 431      1724 1
: 432      1725 1
: 433      1726 1
: 434      1727 1
: 435      1728 1
: 436      1729 2
: 437      1730 2
: 438      1731 2
: 439      1732 2
: 440      1733 2
: 441      1734 1
:          1    FORMAL PARAMETERS:
:          1    NONE
:          1    IMPLICIT INPUTS:
:          1    IMPLICIT OUTPUTS:
:          1    None
:          1    ROUTINE VALUE:
:          1    The contents of module level OWN WAIT
:          1    SIDE EFFECTS:
:          1    None
:          1    --
:          1    BEGIN
:          1    JUST return the value of the module level OWN WAIT
:          1    RETURN .WAIT;
:          1    END;
:          1    !End of BASSREAD_WAIT

```

0000 00000 BASSREAD_WAIT:
 50 00000000' EF 3C 00002 .BORD Save nothing
 04 00C09 MOVZWL WAIT, R0
 RET

: Routine Size: 10 bytes, Routine Base: _BASSCODE + 0020

: 442 1735 1

: 1697
: 1733
: 1734

```

: 444      1736 1 GLOBAL ROUTINE BASSRECOUNT           ! RECOUNT
: 445      1737 1 : =
: 446      1738 1
: 447      1739 1 ++
: 448      1740 1 FUNCTIONAL DESCRIPTION:
: 449      1741 1
: 450      1742 1 This routine supports the Basic RECOUNT function. It returns the number
: 451      1743 1 of bytes read on the last Get. It utilizes a piece of OWN storage which
: 452      1744 1 is written to by the record processing levels which do Gets. In order
: 453      1745 1 to keep the OWN storage from having to be global, this routine is included
: 454      1746 1 in this module.
: 455      1747 1
: 456      1748 1 FORMAL PARAMETERS:
: 457      1749 1     NONE
: 458      1750 1
: 459      1751 1 IMPLICIT INPUTS:
: 460      1752 1     RECOUNT.rl          The number of bytes read on the last GET
: 461      1753 1
: 462      1754 1 IMPLICIT OUTPUTS:
: 463      1755 1     NONE
: 464      1756 1
: 465      1757 1
: 466      1758 1
: 467      1759 1
: 468      1760 1 ROUTINE VALUE:
: 469      1761 1     NUM_OF_BYTES.wl.v    number of bytes read on last Get
: 470      1762 1
: 471      1763 1
: 472      1764 1 SIDE EFFECTS:
: 473      1765 1
: 474      1766 1     --
: 475      1767 1
: 476      1768 2     BEGIN
: 477      1769 2     RETURN .RECOUNT
: 478      1770 1     END;                      ! End of BASSRECOUNT

```

50 00000000' EF 0000 0000 00 0002 04 00009	.ENTRY BASSRECOUNT. Save nothing MOVL RECOUNT, R0 RET
--	---

; Routine Size: 10 bytes, Routine Base: _BASSCODE + 002A

; 479 1771 1

: 1736
: 1769
: 1770

```

481      1772 1 GLOBAL ROUTINE BASS$RECOU_INIT : NOVALUE =      ! Initialize RECOUNT
482      1773 1
483      1774 1 ++
484      1775 1 FUNCTIONAL DESCRIPTION:
485      1776 1
486      1777 1
487      1778 1 This routine initializes the RECOUNT variable. It is used before a RUN
488      1779 1 compiler command in case the previous run of the user's program left
489      1780 1 something in RECOUNT.
490      1781 1
491      1782 1 FORMAL PARAMETERS:
492      1783 1
493      1784 1     NONE
494      1785 1
495      1786 1 IMPLICIT INPUTS:
496      1787 1
497      1788 1     NONE
498      1789 1
499      1790 1 IMPLICIT OUTPUTS:
500      1791 1
501      1792 1     RECOUNT.wL Always set to zero.
502      1793 1
503      1794 1
504      1795 1
505      1796 1     NONE
506      1797 1
507      1798 1
508      1799 1
509      1800 1
510      1801 1
511      1802 2
512      1803 2     BEGIN
513      1804 1     RECOUNT = 0;
                      END;                                ! End of BASS$RECOU_INIT

```

00000000' EF 0000 00000
 D4 00002
 04 00008

.ENTRY BASS\$RECOU_INIT, Save nothing
 CLRL RECOUNT
 RET

: Routine Size: 9 bytes, Routine Base: _BASS\$CODE + 0034

: 514 1805 1

: 1772
 : 1803
 : 1804

```
516      1806 1 GLOBAL ROUTINE BASS$BLNK_LINE (
517          1607 1     FORMAT_CHAR) : CALL_CCB NOVALUE =           ! write a blank line
518          1808 1
519          1809 1     ++
520          1810 1     FUNCTIONAL DESCRIPTION:
521          1811 1
522          1812 1     Print out a blank line. This is needed between arrays.
523          1813 1
524          1814 1     FORMAL PARAMETERS:
525          1815 1
526          1816 1     FORMAT_CHAR.rlu.v           the format character last used
527          1817 1
528          1818 1     IMPLICIT INPUTS:
529          1819 1
530          1820 1     NONE
531          1821 1
532          1822 1     IMPLICIT OUTPUTS:
533          1823 1
534          1824 1     NONE
535          1825 1
536          1826 1     COMPLETION CODES:
537          1827 1
538          1828 1     NONE
539          1829 1
540          1830 1     SIDE EFFECTS:
541          1831 1
542          1832 1     NONE
543          1833 1
544          1834 1     --
545          1835 1
546          1836 2     BEGIN
547          1837 2
548          1838 2     EXTERNAL REGISTER
549          1839 2     CCB : REF BLOCK [. BYTE];
550          1840 2
551          1841 2     LOCAL
552          1842 2     RMS_STATUS;
553          1843 2
554          1844 2
555          1845 2     Actually put out the blank line here.
556          1846 2
557          1847 2     CCB [RABSU_RSZ] = 0;
558          1848 2     CCB [LUBSA_BUF_PTR] = .CCB [LUBSA_BUF_BEG];
559          1849 2     CCB [LUB$A_BAS_VFC1] = BASSK_LF;
560          1850 2     CCB [LUB$B_BAS_VFC2] = BASSK_CR;
561          1851 2
562          1852 2     RMS_STATUS = SPUT (RAB = .CCB);
563          1853 2
564          1854 2     IF .RMS_STATUS EQL RMSS_CONTROLC
565          1855 2     THEN
566          1856 2     BASS$SIGNAL_CTRLC ();
567          1857 2
568          1858 2     IF NOT .RMS_STATUS
569          1859 2     THEN
570          1860 2     PUT_ERROR (K_STOP);
571          1861 2
572          1862 2     RETURN;
```

: 573 1863 1 END;

!End of BASSBLNK_LINE

				.EXTRN	SYSSPUT	
				.ENTRY	BASSBLNK_LINE, Save R2	: 1806
				CLRW	#34(CC8)	: 1847
				MOVL	-68(CC8), -80(CC8)	: 1848
				MOVW	#36097, -38(CC8)	: 1849
				PUSHL	CCB	: 1852
				CALLS	#1. SYSSPUT	: 1854
				MOVL	R0, RMS STATUS	: 1856
				CMPBL	RMS_STATUS, #67153	: 1858
				BNEQ	1\$: 1860
				CALLS	#0, BASS\$SIGNAL_CTRLC	: 1863
				BLBS	RMS_STATUS, 2\$	
				CLRL	-(SP)	
				CALLS	#1, PUT_ERROR	
				RET		

: Routine Size: 55 bytes, Routine Base: _BASSCODE + 003D

: 574 1864 1

576 1865 1 GLOBAL ROUTINE BASS\$REC_MPR1 ! Write Mat Print record
577 1866 1 : JSB_REC1 NOVALUE =
578 1867 1
579 1868 1 ++
580 1869 1 FUNCTIONAL DESCRIPTION:
581 1870 1
582 1871 1 Write one sequential formatted record and initialize for the next
583 1872 1 BASS\$REC_MPR1 writes one record for 10 MAT PRINT A() and then
584 1873 1 initializes the output buffer and returns start and end+1 of user
585 1874 1 part of record buffer to be filled by caller.
586 1875 1 FLR records are space padded.
587 1876 1
588 1877 1 FORMAL PARAMETERS:
589 1878 1
590 1879 1 NONE
591 1880 1
592 1881 1 IMPLICIT INPUTS:
593 1882 1
594 1883 1 LUB\$V_FORM_CHAR =1, comma or semicolon format character
595 1884 1 LUB\$W_RBUF_SIZE Size (bytes) allocated for record buffer at OPEN.
596 1885 1 LUBSA_RBUF_ADR Address of record buffer from OPEN
597 1886 1 LUBSA_BUF_END points to last char inserted into buffer
598 1887 1 by UDF level I/O.
599 1888 1 LUB\$V_FORCEABLE Indicates a forcible device
600 1889 1 LUB\$V_OUTBUF_DR Indicates that there is valid data in the output
601 1890 1 buffer
602 1891 1 RABSW_RSZ Record size
603 1892 1
604 1893 1 IMPLICIT OUTPUTS:
605 1894 1
606 1895 1 LUB\$B_BAS_VFC2 'Post' carriage control for terminal devices
607 1896 1 LUBSA_BUF_PTR Address of next char in user part
608 1897 1 of record buffer
609 1898 1 LUBSA_BUF_END Address of last+1 char in user part
610 1899 1 of record buffer
611 1900 1 LUB\$V_OUTBUF_DR indicates valid data in the output buffer
612 1901 1 LUBSA_BUF_BEG Beginning of the user buffer
613 1902 1 RABSL_RBF Pointer to the user record buffer.
614 1903 1
615 1904 1 ROUTINE VALUE:
616 1905 1
617 1906 1 NONE
618 1907 1
619 1908 1 SIDE EFFECTS:
620 1909 1
621 1910 1 NONE
622 1911 1 --
623 1912 1
624 1913 2 BEGIN
625 1914 2
626 1915 2 EXTERNAL REGISTER
627 1916 2 (CB : REF BLOCK [, BYTE]);
628 1917 2
629 1918 2
630 1919 2 LOCAL RMS_STATUS;
631 1920 2
632 1921 2 !+

```

633      1922 2      | If there is no format character, then set the 'pre' and 'post'
634      1923 2      | carriage control to delimit a record.
635      1924 2
636      1925 2
637      1926 2
638      1927 2
639      1928 3      IF NOT .CCB [LUBSV_FORM_CHAR]
640      1929 3      THEN
641      1930 3      BEGIN
642      1931 2      CCB [LUBSB_BAS_VFC1] = BASSK_LF;
643      1932 2      CCB [LUBSB_BAS_VFC2] = BASSK_CR;
644      1933 2      END;
645      1934 2      +
646      1935 2      Set recordsize to actual length of record
647      1936 2
648      1937 2      -
649      1938 2
650      1939 2
651      1940 2      CCB [RABSW_RSZ] = .CCB [LUBSA_BUF_PTR] - .CCB [LUBSA_BUF_BEG];
652      1941 2
653      1942 2      +
654      1943 2      Output buffer to RMS and check for errors
655      1944 2      | If errors, SIGNAL BASS_FATSYSIO (12='FATAL SYSTEM I/O FAILURE')
656      1945 2
657      1946 2
658      1947 2
659      1948 2
660      1949 2
661      1950 2      IF .RMS_STATUS EQL RMSS_CONTROLC
662      1951 2      THEN
663      1952 2      BASS$SIGNAL_CTRLC ();
664      1953 2
665      1954 2
666      1955 2
667      1956 2
668      1957 2      +
669      1958 2      | Not OPEN or CONNECT - RMS record operation
670      1959 2
671      1960 2
672      1961 2
673      1962 2
674      1963 2      |
675      1964 2      Return next output buffer start and end addresses
676      1965 2
677      1966 2      CCB [LUBSA_BUF_PTR] = .CCB [LUBSA_RBUF_ADR];
678      1967 2      CCB [LUBSA_BUF_END] = .CCB [LUBSA_RBUF_ADR] + .CCB [LUBSW_RBUF_SIZE];
679      1968 2      RETURN;
680      1969 1      END;
                                         ! End of routine - BASS$REC_MPR1

```

06	FE DA	AB AB	8D01	02 8F	DD 00002 BO 00007	BAS\$REC_MPR1::	PUSHL	R2
							BBS	#2
							MOVW	-2(CC8)
								1\$
								#36097, -38(CC8)

: 1865
: 1926
: 1929

22 AB	B0 AB	BC	AB A3 0000D	18:	SUBW3	-68(CC(B), -80(CC(B), 34(CC(B)	1937
	28 AB		D0 00014		MOVL	-20(CC(B), 40(CC(B)	1944
	FE AB	EC	08 BA 00019		BICB2	#8, -2(CC(B)	1945
	00000000G 00		5B DD 0001D		PUSHL	CC(B	1947
	52		01 FB 0001F		CALLS	#1, SYSSPUT	
00010651	8F		50 D0 00026		MOVL	R0, RMS STATUS	
	00000000G 00		52 D1 00029		CMPL	RMS_STATUS, #67153	1949
	07		07 12 00030		BNEQ	28	
	0000V CF		00 FB 00032	28:	CALLS	#0, BAS\$SIGNAL_CTRLC	1951
	B0 AB	EC	52 E8 00039		BLBS	RM\$ STATUS, 38	1953
	50		7E D4 0003C		CLRL	-(SP)	1960
	B4 AB	BB40	01 FB 0003E	38:	CALLS	#1 PUT ERROR	
			D0 00043		MOVL	-20(CC(B), -80(CC(B)	1966
			D2 AB 3C 00048		MOVZWL	-46(CC(B) R0	1967
			9E 0004C		MOVAB	#-20(CC(B)[R0], -76(CC(B)	
			04 BA 00052		POPR	#^M<R2>	
			05 00054		RSB		1969

; Routine Size: 85 bytes, Routine Base: _BASSCODE + 0074

; 681 1970 1

```

683   1971 1 GLOBAL ROUTINE BASSREC_MPR9           ! Mat Write sequential
684   1972 1 : JSB_REC9 NOVALUE =
685   1973 1
686   1974 1 ++
687   1975 1 FUNCTIONAL DESCRIPTION:
688   1976 1
689   1977 1 This routine does not write a record. Presumably the MAT PRINT element
690   1978 1 transmitter took care of all of that. Since we do not want a blank line
691   1979 1 after the array, there is no need to write anything here.
692   1980 1
693   1981 1 FORMAL PARAMETERS:
694   1982 1     NONE
695   1983 1
696   1984 1
697   1985 1
698   1986 1
699   1987 1 IMPLICIT INPUTS:
700   1988 1     LUBSW_RBUF_SIZE      Size (bytes) allocated for record buffer at OPEN.
701   1989 1     LUBSA_RBUF_ADR       Address of record buffer from OPEN
702   1990 1
703   1991 1 IMPLICIT OUTPUTS:
704   1992 1     LUBSA_BUF_PTR        Address of next char in user part
705   1993 1     of record buffer
706   1994 1     LUBSA_BUF_END         Address of last+1 char in user part
707   1995 1     of record buffer
708   1996 1
709   1997 1 ROUTINE VALUE:
710   1998 1     NONE
711   1999 1
712   2000 1
713   2001 1 SIDE EFFECTS:
714   2002 1     NONE
715   2003 1
716   2004 1
717   2005 1
718   2006 2 BEGIN
719   2007 2
720   2008 2 EXTERNAL REGISTER
721   2009 2     CCB : REF BLOCK [, BYTE];
722   2010 2
723   2011 2
724   2012 2 ++
725   2013 2     ! Return next output buffer start and end addresses
726   2014 2
727   2015 2     !-
728   2016 2     CCB [LUBSA_BUF_PTR] = .[CCB [LUBSA_RBUF_ADR];
729   2017 2     CCB [LUBSA_BUF_END] = .[CCB [LUBSA_RBUF_ADR] + .CCB [LUBSW_RBUF_SIZE];
730   2018 1     RETURN;
                           ! END BASSREC_MPR9
END;

```

B0 AB	EC AB	DO 00000 BASSREC_MPR9::	MOVL -20(CCB), -80(CCB)
B4 AB	D2 AB	3C 00005	MOVZWL -46(CCB), R0
	EC BB40	9E 00009	MOVAB a-20(CCB)[R0], -76(CCB)

: 2015
: 2016

BASS\$REC_PROC
1-095

L⁵
16-Sep-1984 01:01:12
14-Sep-1984 11:56:35

VAX-11 Bliss-32 V4.0-742
[BASRTL.SRC]BASREC(PO.B32;1

Page 19
(9)

05 0000F RSB

; Routine Size: 16 bytes. Routine Base: _BASS\$CODE + 00C9

; 731 2019 1

; 2018

```

733 2020 1 GLOBAL ROUTINE BASS$REC_WSL9           ! Write sequential formatted
734 2021 1 : JSB_REC9 NOVALUE =
735 2022 1
736 2023 1
737 2024 1
738 2025 1
739 2026 1
740 2027 1
741 2028 1
742 2029 1
743 2030 1
744 2031 1
745 2032 1
746 2033 1
747 2034 1
748 2035 1
749 2036 1
750 2037 1
751 2038 1
752 2039 1
753 2040 1
754 2041 1
755 2042 1
756 2043 1
757 2044 1
758 2045 1
759 2046 1
760 2047 1
761 2048 1
762 2049 1
763 2050 1
764 2051 1
765 2052 1
766 2053 1
767 2054 1
768 2055 1
769 2056 1
770 2057 1
771 2058 1
772 2059 1
773 2060 1
774 2061 1
775 2062 1
776 2063 1
777 2064 1
778 2065 1
779 2066 1
780 2067 1
781 2068 1
782 2069 1
783 2070 1
784 2071 1
785 2072 1
786 2073 1
787 2074 1
788 2075 1
789 2076 2
    
```

GLOBAL ROUTINE BASS\$REC_WSL9 ! Write sequential formatted
 : JSB_REC9 NOVALUE =

FUNCTIONAL DESCRIPTION:

Write one sequential formatted record and initialize for the next BASS\$REC_WSL1 (and BASS\$REC_WSL9) writes one output buffer and then initializes the output buffer and returns start and end+1 of user part of record buffer to be filled by caller.
 FLR records are space padded.
 /logical record number is incremented/.

FORMAL PARAMETERS:

NONE

IMPLICIT INPUTS:

ISBSV_PRINT_INI	flag to indicate whether there was an element transmitter
LUBSW_RBUF_SIZE	Size (bytes) allocated for record buffer at OPEN.
LUBSA_RBUF_ADR	Address of record buffer from OPEN
LUBSA_BUF_END	points to last char inserted into buffer by UDF level I/O.
LUBSV_FORM_CHAR	The last element transmitter ended in a comma or semicolon format char,
LUBSV_FORCEABLE	Indicates a forcible device
LUBSV_OUTBUF_DR	Indicates that there is valid data in the output buffer
RABSW_RSZ	Record size

IMPLICIT OUTPUTS:

ISBSV_PRINT_INI	reset flag
LUBSB_BAS_VFC2	'Post' carriage control for terminal devices
LUBSA_BUF_PTR	Address of next char in user part of record buffer
LUBSA_BUF_END	Address of last+1 char in user part of record buffer
LUBSV_OUTBUF_DR	indicates valid data in the output buffer
LUBSA_BUF_BEG	Beginning of the user buffer
RABSL_RBF	Pointer to the user record buffer.

ROUTINE VALUE:

NONE

SIDE EFFECTS:

NONE

--

BEGIN

EXTERNAL REGISTER
 CCB : RFF BLOCK [, BYTE];

```
790      2077 2
791      2078 2
792      2079 2
793      2080 2
794      2081 2
795      2082 2
796      2083 2
797      2084 2
798      2085 2
799      2086 2
800      2087 2
801      2088 2
802      2089 2
803      2090 2
804      2091 2
805      2092 2
806      2093 2
807      2094 2
808      2095 2
809      2096 2
810      2097 2
811      2098 2
812      2099 2
813      2100 2
814      2101 2
815      2102 2
816      2103 2
817      2104 2
818      2105 2
819      2106 2
820      2107 2
821      2108 2
822      2109 2
823      2110 2
824      2111 2
825      2112 2
826      2113 2
827      2114 2
828      2115 2
829      2116 2
830      2117 2
831      2118 2
832      2119 2
833      2120 2
834      2121 2
835      2122 2
836      2123 2
837      2124 2
838      2125 2
839      2126 2
840      2127 2
841      2128 2
842      2129 2
843      2130 2
844      2131 2
845      2132 2
846      2133 2

    LOCAL
        RMS_STATUS;

    [+]
    | If last element ended with a format character and not a terminal device
    | then return to caller without writing anything. With CR format, we must
    | PUT a whole record.
    |-]

    IF .CCB [LUB$V_FORM_CHAR] AND NOT .CCB [LUB$V_FORCE] THEN RETURN;

    [+]
    | Set the 'post' carriage control to carriage return
    | if the last element transmitter had no format character following.
    |-]

    IF NOT .CCB [LUB$V_FORM_CHAR] THEN CCB [LUB$B_BAS_VFC2] = BASSK_CR;

    [+]
    | Set recordszie to actual length of record
    |-]

    CCB [RAB$W_RSZ] = .CCB [LUB$A_BUF_PTR] - .CCB [LUB$A_BUF_BEG];

    [+]
    | Output buffer to RMS and check for errors
    | If errors, SIGNAL BASS_FATSYSIO (12='FATAL SYSTEM I/O FAILURE')
    |-]

    CCB [RAB$L_RBF] = .CCB [LUB$A_BUF_BEG];

    [+]
    | Write something if there is something in the buffer or if there was no
    | element transmitter.
    |-]

    IF .CCB [LUB$V_OUTBUF_DR] OR .CCB [ISB$V_PRINTINI]
    THEN

        RMS_STATUS = $PUT (RAB = .CCB);

        IF .RMS_STATUS EQL RMSS_CONTROLC
        THEN
            BAS$SIGNAL_CTRLC ();

        IF NOT .RMS_STATUS
        THEN
            PUT_ERROR (K_STOP);

        CCB [LUB$V_OUTBUF_DR] = 0;
        CCB [ISB$V_PRINTINI] = 0;

    [+]
    | If there is no format character then set the 'pre' carriage control to LF
    | for the next record. This is recursive I/O and the rest of the list when
    | we return should be written on the next line.
    |-]
```

```

: 847    2134 2      IF NOT .CCB [LUB$V_FORM_CHAR] THEN CCB [LUB$B_BAS_VFC1] = BASSK_LF;
: 848    2135 2
: 849    2136 2
: 850    2137 2      !+
: 851    2138 2      | Return next output buffer start and end addresses
: 852    2139 2      !-
: 853    2140 2      CCB [LUB$A_BUF_PTR] = .CCB [LUB$A_RBUF_ADR];
: 854    2141 2      CCB [LUB$A_BUF_END] = .CCB [LUB$A_RBUF_ADR] + .CCB [LUB$W_RBUF_SIZE];
: 855    2142 2      RETURN;
: 856    2143 1      END;                                ! END OF ROUTINE

```

				52 DD 00000 BASS\$REC_WSL9::			
0A	FE AB			02 E1 00002	PUSHL	R2	2020
66	FE AB			06 E1 00007	BBC	#2, -2(CC(B), 1\$	2087
05	DB AB			02 E0 0000C	BBS	#6, -2(CC(B), 8\$	2094
22	AB	8D	8F	90 00011 1\$: AB	MOVBL	#2, -2(CC(B), 2\$	
	BO AB	BC	AB	00016 2\$: D0	SUBW3	#-115, -37(CC(B))	
05	28 AB	BC	AB	00022 3\$: E0	MOVL	-68(CC(B), -80(CC(B), 34(CC(B)	2100
0C	FE AB		03	00027 3\$: E1	BBS	-68(CC(B), 40(CC(B)	2107
	97 AB		5B	0002C 3\$: DD	BBC	#3, -2(CC(B), 3\$	2113
			01	0002E 3\$: FB	PUSHL	#3, -105(CC(B), 4\$	
	00000000G 00		50	00035 3\$: 00	CALLS	CC(B	2116
	52		52	00038 4\$: D0	MOVL	#1, SY\$PUT	
	00010651 8F		52	00038 4\$: D1	CMLP	R0, RMS_STATUS	
			07	12 0003F 5\$: 00	BNEQ	RMS_STATUS, #67153	2118
	00000000G 00		00	FB 00041 5\$: FB	CALLS	5\$	
	07		52	00048 5\$: E8	BLBS	#0, BASS\$SIGNAL_CTRL_C	2120
			7E	0004B 5\$: D4	CLRL	RMS_STATUS, 6\$	2122
	0000V CF		01	0004D 6\$: FB	CALLS	-(SP)	2124
	FE AB		08	00052 6\$: 8A	BICB2	#1, PUT_ERROR	
	97 AB		08	00056 6\$: 8A	BICB2	#8, -2(CC(B)	2126
04	FE AB		02	0005A 6\$: E0	BBS	#8, -105(CC(B)	2127
	DA AB		01	0005F 7\$: 90	MOVBL	#2, -2(CC(B), 7\$	2134
	80 AB	EC	AB	00063 7\$: D0	MOVL	#1, -38(CC(B)	
	50	D2	AB	3C 00068 7\$: 3C	MOVZWL	-20(CC(B), -80(CC(B)	2140
	B4 AB	EC BB40	9E	0006C 7\$: 9E	MOVAB	-46(CC(B), R0	2141
			04	00072 8\$: BA	POPR	0-20(CC(B)[R0], -76(CC(B)	
			05	00074 8\$: 05	RSB	#^M<R2>	2143

: Routine Size: 117 bytes, Routine Base: _BASS\$CODE + 00D9

: 857 2144 1

```

: 859      2145 1 GLOBAL ROUTINE BASS$REC_RSLO          ! Read initialization
: 860      2146 1 : JSB_REC0 NOVALUE =
: 861      2147 1
: 862      2148 1 ++
: 863      2149 1 // FUNCTIONAL DESCRIPTION:
: 864      2150 1
: 865      2151 1 BASS$REC_RSLO (and BASS$REC_RSF1) reads one record if this is not a terminal.
: 866      2152 1 Then return start and end+1 of user
: 867      2153 1 part of record to be processed as input.
: 868      2154 1
: 869      2155 1 // FORMAL PARAMETERS:
: 870      2156 1
: 871      2157 1     NONE
: 872      2158 1
: 873      2159 1 // IMPLICIT INPUTS:
: 874      2160 1
: 875      2161 1     LUBSW_RBUF_SIZE           Size of record buffer allocated in OPEN.
: 876      2162 1     LUBSA_RBUF_ADR            Address of record buffer from OPEN.
: 877      2163 1     LUBSV_TERM_DEV           flag in LUB which indicates a terminal device.
: 878      2164 1     RABSW_RSZ                word in the RAB which contains the buffer size.
: 879      2165 1     RABSL_RBF                longword in RAB which points to the buffer.
: 880      2166 1     LUBSL_WAIT_TIME         Max time to wait for input, in seconds.
: 881      2167 1     WAIT                    The module level OWN WAIT
: 882      2168 1
: 883      2169 1 // IMPLICIT OUTPUTS:
: 884      2170 1
: 885      2171 1     RECOUNT                 Global storage to hold number of bytes read from
: 886      2172 1
: 887      2173 1     LUBSL_LOG_RECNO        last input.
: 888      2174 1
: 889      2175 1     LUBSA_BUF_PTR          Increment logical record number
: 890      2176 1
: 891      2177 1     LUBSA_BUF_END          of next record to be read.
: 892      2178 1
: 893      2179 1
: 894      2180 1 // ROUTINE VALUE:
: 895      2181 1
: 896      2182 1     NONE
: 897      2183 1
: 898      2184 1 // SIDE EFFECTS:
: 899      2185 1
: 900      2186 1     Reads next record from file on this logical unit.
: 901      2187 1     Throws away things that are pending in the Print buffer for non-terminal
: 902      2188 1     devices.
: 903      2189 1     SIGNALS RMS errors directly.
: 904      2190 1     SIGNALS BASSK_TIMLIMEXC if a wait time was specified and we
: 905      2191 1     exceed it.
: 906      2192 1 // --
: 907      2193 1
: 908      2194 2 // BEGIN
: 909      2195 2
: 910      2196 2 // EXTERNAL REGISTER
: 911      2197 2   CCB : REF BLOCK [, BYTE];
: 912      2198 2
: 913      2199 2 // LITERAL
: 914      2200 2   K_ESCAPE = %X'1B';
: 915      2201 2   K_CR = %X'0D';

```

```
916      2202 2
917      2203 2
918      2204 2
919      2205 2
920      2206 2
921      2207 2
922      2208 2
923      2209 2
924      2210 2
925      2211 2
926      2212 2
927      2213 2
928      2214 2
929      2215 2
930      2216 2
931      2217 2
932      2218 2
933      2219 2
934      2220 2
935      2221 2
936      2222 2
937      2223 2
938      2224 2
939      2225 2
940      2226 2
941      2227 2
942      2228 2
943      2229 2
944      2230 2
945      2231 2
946      2232 2
947      2233 2
948      2234 2
949      2235 2
950      2236 2
951      2237 2
952      2238 2
953      2239 2
954      2240 2
955      2241 2
956      2242 2
957      2243 2
958      2244 2
959      2245 2
960      2246 2
961      2247 2
962      2248 2
963      2249 2
964      2250 2
965      2251 2
966      2252 2
967      2253 2
968      2254 2
969      2255 2
970      2256 4
971      2257 4
972      2258 4

        LOCAL
          RMS_STATUS,
          WAIT_TIME;
          ! Current wait time

        If a timeout has been specified, store information in the RAB to tell
        RMS about it. If no timeout has been specified, clear the TMO bit
        in case there was an earlier timeout specified.

        If WAIT is zero then use the LUB's wait. This is to provide upward compatibility
        i.e. existing EXE's can run with the LUB wait value in V2.2.

        WAIT_TIME = ( IF ( .WAIT EQL 0 ) THEN .CCB [ LUBSL_WAIT_TIME ] ELSE .WAIT );
        IF ( .WAIT_TIME EQL 0 )
        THEN
          CCB [RABSV_TMO] = 0
        ELSE
          BEGIN
            CCB [RABSB_TMO] = .WAIT_TIME;
            CCB [RABSV_TMO] = 1;
          END;

        Set the Read-no-echo RMS bit based on the user's last call to
        ECHO or NOECHO.

        CCB [RABSV_RNE] = .CCB [LUBSV_NOECHO];

        Check to see if this is a terminal device. If this is NOT
        a terminal then do a GET. GETs for terminals are done each time more
        data are needed.
        Read record into buffer using RMS and check for errors

        IF ( NOT .CCB [LUBSV_TERM_DEV] OR .CCB [LUBSV_ANSI] )
        THEN
          BEGIN
            LOCAL
              TEMP_CCB : REF BLOCK [ , BYTE];      ! Temporary CCB
              TEMP_CCB = .CCB [LUBSA_BUDDY_PTR];
              If there is something pending in the Print buffer, then SPUT it.
              It cannot become a prompt, because RMS will throw away prompts
              to disk files; therefore we must SPUT it.

              IF (NOT .CCB [LUBSV_TERM_DEV]) AND .TEMP_CCB [LUBSV_OUTBUF_DR]
              THEN
                BEGIN
                  TEMP_CCB [RABSW_RSZ] = .TEMP_CCB [LUBSA_BUF_PTR] - .TEMP_CCB [LUBSA_BUF_BEG];
                  TEMP_CCB [RABSL_RBF] = .TEMP_CCB [LUBSA_BUF_BEG];
                END;
          END;
```

```
973      2259   6
974      2260   4
975      2261   4
976      2262   4
977      2263   4
978      2264   4
979      2265   4
980      2266   4
981      2267   4
982      2268   4
983      2269   3
984      2270   3
985      2271   3
986      2272   3
987      2273   3
988      2274   3
989      2275   3
990      2276   3
991      2277   3
992      2278   3
993      2279   3
994      2280   3
995      2281   3
996      2282   3
997      2283   3
998      2284   3
999      2285   3
1000     2286   3
1001     2287   3
1002     2288   3
1003     2289   3
1004     2290   3
1005     2291   3
1006     2292   3
1007     2293   3
1008     2294   3
1009     2295   3
1010     2296   3
1011     2297   3
1012     2298   3
1013     2299   3
1014     2300   3
1015     2301   3
1016     2302   3
1017     2303   3
1018     2304   3
1019     2305   3
1020     2306   3
1021     2307   2
1022     2308   2
1023     2309   2
1024     2310   2
1025     2311   2
1026     2312   2
1027     2313   2
1028     2314   2
1029     2315   2

        RMS_STATUS = SPUT (RAB = .TEMP_CCB);
        IF .RMS_STATUS EQL RMSS_CONTROLC
        THEN
            BAS$SIGNAL_CTRLC ();
        IF NOT .RMS_STATUS
        THEN
            PUT_ERROR (K_STOP);
        END;

        TEMP_CCB [LUBSA_BUF_PTR] = .TEMP_CCB [LUBSA_BUF_BEG];
        RMS_STATUS = $GET (RAB = .CCB);
        IF .RMS_STATUS EQL RMSS_CONTROLC
        THEN
            BAS$SIGNAL_CTRLC ();
        IF NOT .RMS_STATUS
        THEN
            GET_ERROR (K_STOP);

        /* Set RECOUNT to the number of bytes read
         * If the file is a terminal format file, then RECOUNT has to be
         * adjusted for the carriage control terminator. Because RMS does not return
         * a terminator for a file, we unconditionally put a CRLF on the end and
         * bump RECOUNT by 2.
        */

        RECOUNT = .CCB [RABSW_RSZ] + (IF (.CCB [LUBSV_TERM_FOR]) AND ((.CCB [LUBSB_RAT] AND FABSM_CR) NEQU 0
        THEN 2 ELSE 0);

        /* Put the CR into the STV field since RMS doesn't
         * We should only do this if the record attributes indicate a CR format.
        */

        IF (.CCB [LUBSB_RAT] AND FABSM_CR) NEQU 0 THEN CCB [RABSL_STV] = 13;

        /* Return start-1 and end+1 address of record just read
        */

        CCB [LUBSA_BUF_PTR] = .CCB [RABSL_RBF] - 1;
        CCB [LUBSA_BUF_END] = .CCB [RABSL_RBF] + .CCB [RABSW_RSZ];
        END
    ELSE
        /* This is a terminal. Force a no data in the buffer condition
         * so the first GET is done on the element transmitter after the
         * Prompt (if any) is known.
        */

        BEGIN
```

```
; 1030    2316 3
; 1031    2317 3
; 1032    2318 2
; 1033    2319 2
; 1034    2320 2
; 1035    2321 1
```

```
CCB [LUBSA_BUF_PTR] = .CCB [RABSL_RBF];
CCB [LUBSA_BUF_END] = .CCB [LUBSA_BUF_PTR];
END;
```

```
RETURN;
END;
```

```
! End of BASS$REC_RSLO
```

.EXTRN SY\$GET

				OC BB 00000 BASS\$REC_RSLO::			
				PUSHR #^M<R2,R3>			2145
				MOVZWL WAIT, R0			2216
				BNEQ 18			
				MOVL -52(CCB), WAIT_TIME			2218
				BNEQ 18			2220
				BICB2 #2, 7(CCB)			
				BRB 28			
				WAIT TIME, 31(CCB)			2223
				#2, 7(CCB)			2224
				-96(CCB), #0, #1, 7(CCB)			2232
				BBC #5, -2(CCB), 38			2241
				BBS #4, -95(CCB), 38			
				BRW 118			
				MOVL -72(CCB), TEMP_CCB			2247
				BBS #5, -2(CCB), 58			2254
				BBC #3, -2(TEMP_CCB), 58			
				SUBW3 -68(TEMP_CCB), -80(TEMP_CCB), 34(TEMP_CCB)			2257
				MOVL -68(TEMP_CCB), 40(TEMP_CCB)			2258
				PUSHL TEMP_CCB			2260
				CALLS #1, SY\$PUT			
				MOV L R0, RMS_STATUS			
				CMP L RMS_STATUS, #67153			2262
				BNEQ 48			
				CALLS #0, BASS\$SIGNAL_CTRLC			2264
				BLBS RMS_STATUS, 58			2266
				CLRL -(SP)			2268
				CALLS #1, PUT_ERROR			
				MOVL -68(TEMP_CCB), -80(TEMP_CCB)			2271
				CCB CCB			2273
				CALLS #1, SY\$GET			
				MOV L R0, RMS_STATUS			
				CMP L RMS_STATUS, #67153			2275
				BNEQ 68			
				CALLS #0, BASS\$SIGNAL_CTRLC			2277
				BLBS RMS_STATUS, 78			2279
				CLRL -(SP)			2281
				CALLS #1, GET_ERROR			
				BBC #4, -2(CCB), 88			2291
				BBC #1, -10(CCB), 88			
				MOVL #2, R0			
				BRB 98			
				CLRL R0			
				MOVZWL 34(CCB), R1			
				ADDL3 R1, R0 RECOUNT			
				BBC #1, -10(CCB), 108			2298

B0 AB	0C AB	0D D0 000C0	01 C3 000C4	10\$:	MOVL #13, 12(CCB)
	28 AB		22 AB 3C 000CA		SUBL3 #1, 40(CCB), -80(CCB)
	50		28 BB40 9E 000CE		MOVZWL 34(CCB) R0
	B4 AB		0A 11 000D4		MOVAB @40(CCB){[R0]}, -76(CCB)
	B0 AB	28 AB	D0 000D6	11\$:	BRB 12\$
	B4 AB	80 AB	D0 000DB		MOVL 40(CCB), -80(CCB)
			OC BA 000E0	12\$:	MOVL -80(CCB){, -76(CCB)}
			05 000E2		POPR #^M[R2,R3]>
					RSB

; Routine Size: 227 bytes, Routine Base: _BASS\$CODE + 014E

; 1036 2322 1

2304
2305
2241
2316
2317
2321

: 1038 2323 1 GLOBAL ROUTINE BASS\$REC_MINO
1039 : JSB_REC0 NOVALUE = ! MAT Input initialization
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094 2324 1
2325 1
2326 1
2327 1
2328 1
2329 1
2330 1
2331 1
2332 1
2333 1
2334 1
2335 1
2336 1
2337 1
2338 1
2339 1
2340 1
2341 1
2342 1
2343 1
2344 1
2345 1
2346 1
2347 1
2348 1
2349 1
2350 1
2351 1
2352 1
2353 1
2354 1
2355 1
2356 1
2357 1
2358 1
2359 1
2360 1
2361 1
2362 1
2363 1
2364 1
2365 1
2366 1
2367 1
2368 1
2369 1
2370 1
2371 1
2372 1
2373 1
2374 2
2375 2
2376 2
2377 2
2378 2
2379 2
GLOBAL ROUTINE BASS\$REC_MINO
: JSB_REC0 NOVALUE =
** FUNCTIONAL DESCRIPTION:
BASS\$REC_RSFO (and BASS\$REC_RSF1) reads one record if this is not a terminal.
Then return start and end+1 of user part of record to be processed as input.
FORMAL PARAMETERS:
NONE
IMPLICIT INPUTS:
LUBSW_RBUF_SIZE Size of record buffer allocated in OPEN.
LUBSA_RBUF_ADR Address of record buffer from OPEN.
LUBSV_TERM_DEV flag in LUB which indicates a terminal device.
RABSW_RSZ Word in the RAB which contains the buffer size.
RABSL_RBF Longword in RAB which points to the buffer.
LUBSL_WAIT_TIME Max time to wait for input, in seconds.
WAIT Module level OWN WAIT
IMPLICIT OUTPUTS:
RECOUNT Global storage to hold number of bytes read from last input.
LUBSL_LOG_RECNO Increment logical record number of next record to be read.
LUBSA_BUF_PTR points to first char of user part of record buffer.
LUBSA_BUF_END points to end+1 of user part of record buffer.
ROUTINE VALUE:
NONE
SIDE EFFECTS:
Reads next record from file on this logical unit.
Throws away things that are pending in the Print buffer for non-terminal devices.
SIGNALS BASSK_FATSYSIO (12='FATAL SYSTEM I/O FAILURE')
SIGNALS BASSK_ENDFILDEV (11='END-OF-FILE ON DEVICE')
SIGNALS BASSK_RECFLTOO if record too big
SIGNALS BASSK_TIMLIMEXC if a wait time was specified and we exceed it.
!--
BEGIN
EXTERNAL REGISTER
[CB : REF BLOCK [, BYTE];
LITERAL

```
: 1095      2380  2      K_ESCAPE = %X'18'.  
: 1096      2381  2      K_R = %X'0D';  
: 1097      2382  2  
: 1098      2383  2  
: 1099      2384  2      LOCAL  
: 1100      2385  2      RMS_STATUS,  
: 1101      2386  2      WAIT_TIME;          !Current wait time  
: 1102      2387  2      If a timeout has been specified, store information in the RAB to tell  
: 1103      2388  2      RMS about it. If no timeout has been specified, clear the TMO bit  
: 1104      2389  2      in case there was an earlier timeout specified.  
: 1105      2390  2  
: 1106      2391  2  
: 1107      2392  2  
: 1108      2393  2  
: 1109      2394  2      If WAIT is zero then use the LUB's wait. This is to provide upward compatibility  
: 1110      2395  2      i.e. existing EXE's can run with the LUB wait value in V2.2.  
: 1111      2396  2      WAIT_TIME = ( IF ( .WAIT EQL 0 ) THEN .CCB [ LUBSL_WAIT_TIME ] ELSE .WAIT );  
: 1112      2397  2      IF (.WAIT_TIME EQL 0)  
: 1113      2398  2      THEN      CCB [ RABSV_TMO ] = 0  
: 1114      2399  2      ELSE      BEGIN  
: 1115      2400  2      CCB [ RABSB_TMO ] = .WAIT_TIME;  
: 1116      2401  2      CCB [ RABSV_TMO ] = 1;  
: 1117      2402  2      END;  
: 1118      2403  2  
: 1119      2404  2  
: 1120      2405  2  
: 1121      2406  2  
: 1122      2407  2  
: 1123      2408  2  
: 1124      2409  2  
: 1125      2410  2      Set the Read-no-echo RMS bit based on the user's last call to  
: 1126      2411  2      ECHO or NOECHO.  
: 1127      2412  2      CCB [ RABSV_RNE ] = .CCB [ LUBSV_NOECHO ];  
: 1128      2413  2  
: 1129      2414  2  
: 1130      2415  2  
: 1131      2416  2      Check to see if this is a terminal device. If this is NOT  
: 1132      2417  2      a terminal then do a GET. GETs for terminals are done each time more  
: 1133      2418  2      data are needed.  
: 1134      2419  2      Read record into buffer using RMS and check for errors  
: 1135      2420  2      If end-of-file, SIGNAL BASSK_ENDFILDEV (11='END-OF-FILE ON DEVICE')  
: 1136      2421  2      If record too big for record-buffer, SIGNAL BASSK_RECVILTOO.  
: 1137      2422  2      If errors, SIGNAL BASSK_FATSYSIO (12='FATAL SYSTEM I/O ERROR')  
: 1138      2423  2  
: 1139      2424  2  
: 1140      2425  3      IF ( NOT .CCB [ LUBSV_TERM_DEV ] )  
: 1141      2426  3      THEN      BEGIN  
: 1142      2427  3  
: 1143      2428  3  
: 1144      2429  3      LOCAL  
: 1145      2430  3      TEMP_CCB : REF_BLOCK [, BYTE];      ! Temporary CCB  
: 1146      2431  3      TEMP_CCB = .CCB [ LUBSA_BUDDY_PTR ];  
: 1147      2432  3  
: 1148      2433  3  
: 1149      2434  3      If there is something pending in the Print buffer, then SPUT it.  
: 1150      2435  3      It cannot become a prompt, because RMS will throw away prompts  
: 1151      2436  3      to disk files; therefore we must SPUT it.
```

```
1152      2437 3
1153      2438 3
1154      2439 3
1155      2440 4
1156      2441 4
1157      2442 4
1158      2443 4
1159      2444 4
1160      2445 4
1161      2446 4
1162      2447 4
1163      2448 4
1164      2449 4
1165      2450 4
1166      2451 4
1167      2452 6
1168      2453 3
1169      2454 3
1170      2455 3
1171      2456 3
1172      2457 3
1173      2458 3
1174      2459 3
1175      2460 3
1176      2461 3
1177      2462 3
1178      2463 3
1179      2464 3
1180      2465 3
1181      2466 3
1182      2467 3
1183      2468 3
1184      2469 3
1185      2470 3
1186      2471 3
1187      2472 3
1188      2473 4
1189      2474 4
1190      2475 4
1191      2476 4
1192      2477 4
1193      2478 3
1194      2479 3
1195      2480 3
1196      2481 3
1197      2482 3
1198      2483 3
1199      2484 3
1200      2485 3
1201      2486 3
1202      2487 3
1203      2488 3
1204      2489 3
1205      2490 3
1206      2491 3
1207      2492 3
1208      2493 3

;-
; IF (NOT .CCB [LUBSV_TERM_DEV]) AND .TEMP_CCB [LUBSV_OUTBUF_DR]
; THEN
; BEGIN
; TEMP_CCB [RABSW_RSZ] = .TEMP_CCB [LUBSA_BUF_PTR] - .TEMP_CCB [LUBSA_BUF_BEG];
; TEMP_CCB [RABSL_RBF] = .TEMP_CCB [LUBSA_BUF_BEG];
; RMS_STATUS = $PUT (RAB = .CCB);
; IF .RMS_STATUS EQL RMSS_CONTROLC
; THEN
;   BAS$SIGNAL_CTRLC ();
; IF NOT .RMS_STATUS
; THEN
;   PUT_ERROR (K_STOP);
; END;
;
; TEMP_CCB [LUBSA_BUF_PTR] = .TEMP_CCB [LUBSA_BUF_BEG];
; RMS_STATUS = $GET (RAB = .CCB);
; IF .RMS_STATUS EQL RMSS_CONTROLC
; THEN
;   BAS$SIGNAL_CTRLC ();
; IF NOT .RMS_STATUS
; THEN
;   GET_ERROR (K_STOP);
;
;+ Set RECOUNT to the number of bytes read
; If the file is a terminal format file, then RECOUNT has to be
; adjusted for the carriage control terminator.
;-
; RECOUNT = .CCB [RABSW_RSZ] + (IF .CCB [LUBSV_TERM_FOR] THEN SELECTONEU .CCB [RABSW_STV0] OF
; SET
; [K_ESCAPE] : .CCB [RABSW_STV2];
; [K_CR] : 2;
; [OTHERWISE] : 0;
; TES ELSE 0);
;
;+ Return start-1 and end+1 address of record just read
;-
; CCB [LUBSA_BUF_PTR] = .CCB [RABSL_RBF] - 1;
; CCB [LUBSA_BUF_END] = .CCB [RABSL_RBF] + .CCB [RABSW_RSZ];
;
;+ Check for an '8' as the last character of the record. If it is there,
; it is a continuation character and signifies that there is more data to
; come in the next record.
;-
; IF .(.CCB [LUBSA_BUF_END] - 1)<0, 8> EOLU K_MAT_CONT_CHAR
```

```

1209      2494  5
1210      2495  4
1211      2496  4
1212      2497  4
1213      2498  4
1214      2499  3
1215      2500  3
1216      2501  3
1217      2502  3
1218      2503  3
1219      2504  3
1220      2505  3
1221      2506  3
1222      2507  3
1223      2508  3
1224      2509  3
1225      2510  3
1226      2511  3
1227      2512  3
1228      2513  3
1229      2514  3
1230      2515  3
1231      2516  2
1232      2517  2
1233      2518  2
1234      2519  1

      THEN
        BEGIN
          CCB [LUBSA_BUF_END] = .CCB [LUBSA_BUF_END] - 1;
          CCB [ISBSV_MAT_CONT] = 1;
        END
      ELSE
        CCB [ISBSV_MAT_CONT] = 0;
      END
    ELSE
      !
      !+ This is a terminal. Force a no data in the buffer condition
      ! so the first GET is done on the element transmitter after the
      ! Prompt (if any) is known. Set the MAT Input continuation flag so that the element
      ! transmitter (REC1) can read the first record.
      !
      BEGIN
        CCB [LUBSA_BUF_PTR] = .CCB [RABSL_RBF];
        CCB [LUBSA_BUF_END] = .CCB [LUBSA_BUF_PTR];
        CCB [ISBSV_MAT_CONT] = 1;
      END;

      RETURN;
    END;
  
```

! End of BASSREC_MINO

OC BB 00000 BASSREC MINO::						
			PUSHR	#MCR2,R3>		
		50 00000000'	EF 3C 00002	MOVZWL	WAIT, R0	
		50 CC	0C 12 00009	BNEQ	1\$	
		07 AB	A8 D0 00008	MOVL	-52(CCB), WAIT_TIME	
			06 12 0000F	BNEQ	1\$	
			02 8A 00011	BICB2	#2, 7(CCB)	
			08 11 00015	BRB	2\$	
		1F AB	50 90 00017	MOV8	WAIT_TIME, 31(CCB)	
		07 AB	02 88 00018	BISB2	#2, 7(CCB)	
		00 00	A0 AB F0 0001F	INSV	-96(CCB), #0, #1, 7(CCB)	
		03 FE	05 E1 00026	BBC	#5, -2(CCB), 38	
		07 AB	00BC 31 00028	BRW	12\$	
		37 FE	B8 AB D0 0002E	MOVL	-72(CCB), TEMP CCB	
		32 A2	05 E0 00032	BBS	#5, -2(CCB), 58	
		22 A2	03 E1 00037	BBC	#3, -2(TEMP CCB), 58	
		28 A2	BC A2 A3 0003C	SUBW3	-68(TEMP CCB), -80(TEMP CCB), 34(TEMP CCB)	
			A2 D0 00043	MOVL	-68(TEMP CCB), 40(TEMP CCB)	
			5B D0 00048	PUSHL	CCB	
		000000006 00	01 FB 0004A	CALLS	#1, SYSSPUT	
		53	50 D0 00051	MOVL	R0, RMS STATUS	
		00010651 BF	53 D1 00054	CMPL	RMS_STATUS, #67153	
			07 12 0005B	BNEQ	4\$	
		00000000G 00	00 FB 0005D	CALLS	#0, BASSSIGNAL_CTRLC	
		07	53 EB 00064	BLBS	RMS_STATUS, 58	
			7E D4 00067	CLRL	-(SP)	

L 6
16-Sep-1984 01:01:12 VAX-11 Bliss-32 v4.0-742
14-Sep-1984 11:56:35 [BASRTL.SRC]BASRECPRO.B32;1

0000V	CF	01	FB	00069	58:	CALLS	#1 PUT ERROR					2455
B0	A2	A2	DO	0006E		MOVL	-68(TEMP_CCB), -80(TEMP_CCB)					2457
00000000G	00	5B	DD	00073		PUSHL	CCB					
	53	01	FB	00075		CALLS	#1. SYSSGET					
00010651	8F	50	DO	0007C		MOVL	R0, RMS STATUS					2459
	53	53	D1	0007F		CMPL	RMS_STATUS, #67153					
00000000G	00	07	12	00086		BNEQ	68					2461
	07	00	FB	00088	68:	CALLS	#0, BASS\$SIGNAL_CTRL[2463
	53	53	E8	0008F		BLBS	RMS STATUS, 78					2465
	7E	D4	00092			CLRL	-(SP)					
19	0000V	CF	01	FB	00094	78:	CALLS	#1. GET_ERROR				2473
	FE	AB	04	E1	00099		BBC	#4 -2(CCB), 98				
	50	AB	3C	0009E		MOVZWL	12(CCB), R0					2475
	1B	50	B1	000A2		CMPW	R0, #27					
	50	06	12	000A5		BNEQ	88					
	50	AB	3C	000A7		MOVZWL	14(CCB), R0					
	0C	0C	11	000AB		BRB	10\$					
	0D	50	B1	000AD	88:	CMPW	R0, #13					2476
	05	50	12	000B0		BNEQ	98					
	50	02	D0	000B2		MOVL	#2, R0					
	50	02	11	000B5		BRB	10\$					
	50	50	D4	000B7	98:	CLRL	R0					2473
	S1	22	AB	3C	000B9	10\$:	MOVZWL	34(CCB), R1				
00000000	EF	50	51	C1	000BD		ADDL3	R1, R0, RECOUNT				
B0	AB	28	AB	01	C3	000C5	SUBL3	#1, 40(CCB), -80(CCB)				2484
	50	22	AB	3C	000CB		MOVZWL	34(CCB), R0				2485
B4	AB	28	BB40	9E	000CF		MOVAB	040(CCB)[R0], -76(CCB)				2493
	50	B4	AB	D0	000D5		MOVL	-76(CCB), R0				
	26	FF	A0	91	000D9		CMPB	-1(R0), #38				
			05	12	000DD		BNEQ	11\$				
			B4	AB	D7	000DF	DECL	-76(CCB)				2496
				10	11	000E2	BRB	13\$				2497
97	AB	02	8A	000E4	118:	BICB2	#2, -105(CCB)					2500
		0E	11	000E8		BRB	14\$					
B0	AB	28	AB	DO	000EA	128:	MOVL	40(CCB), -80(CCB)				2513
B4	AB	B0	AB	DO	000EF		MOVL	-80(CCB), -76(CCB)				2514
97	AB	02	88	000F4	138:	BISB2	#2, -105(CCB)					2515
		0C	BA	000F8	148:	POPR	#^M<R2,R3>					2519
			05	000FA		RSB						

; Routine Size: 251 bytes, Routine Base: _BASSCODE + 0231

: 1235 2520 1

1294 2578 2 | data are needed. If this is not a terminal device then error.
1295 2579 2 | Read record into buffer using RMS and check for errors
1296 2580 2
1297 2581 2
1298 2582 2
1299 2583 2
1300 2584 2
1301 2585 2
1302 2586 2
1303 2587 2
1304 2588 2
1305 2589 2
1306 2590 2
1307 2591 2
1308 2592 2
1309 2593 2
1310 2594 2
1311 2595 2
1312 2596 2
1313 2597 2
1314 2598 2
1315 2599 2
1316 2600 2
1317 2601 2
1318 2602 2
1319 2603 2
1320 2604 2
1321 2605 2
1322 2606 2
1323 2607 2
1324 2608 2
1325 2609 2
1326 2610 2
1327 2611 2
1328 2612 2
1329 2613 2
1330 2614 2
1331 2615 2
1332 2616 2
1333 2617 2
1334 2618 2
1335 2619 2
1336 2620 2
1337 2621 2
1338 2622 2
1339 2623 2
1340 2624 2
1341 2625 2
1342 2626 2
1343 2627 2
1344 2628 2
1345 2629 2
1346 2630 2
1347 2631 2
1348 2632 2
1349 2633 2
1350 2634 2

| IF (NOT .CCB [LUB\$V_ANSI]) AND .CCB [LUB\$V_TERM_DEV]
| THEN BEGIN
| RMS_STATUS = \$GET (RAB = .CCB);
| IF .RMS_STATUS EQL RMSS_CONTROLC
| THEN BAS\$SIGNAL_CTRLC ();
| IF NOT .RMS_STATUS
| THEN GET_ERROR (K_STOP);

|+ Return start-1 and end+1 address of record just read
| LUBSA_BUF_PTR is set to the beginning-1 of the buffer only for BASIC
| Input. This is seen as a solution to the problem of the user entering
| <return> as the response to a prompt (null input record) and an empty
| or depleted buffer which requires another Get.
| The algorithm:
| 1) Does LUBSA_BUF_PTR = LUBSA_BUF_END?
| T: The buffer is depleted = another Get is required.
| 2) Add one to LUBSA_BUF_PTR
| 3) Does LUBSA_BUF_PTR = LUBSA_BUF_END?
| T: Return the default value.
| 4) Scan for the next field.

| [CB [LUBSA_BUF_PTR] = .[CB [RAB\$L_RBF] - 1;
| [CB [LUBSA_BUF_END] = .[CB [RAB\$L_RBF] + .[CB [RAB\$W_RSZ];
| END
| ELSE

|+ This is not a terminal device
| Signal insufficient data unless this is an ANSI INPUT.
| ANSI INPUT errors should cause the statement to be restarted.
| (This happens in BAS\$SHANDLER).

| IF NOT .CCB [LUB\$V_ANSI]
| THEN BAS\$SIGNAL (BASSK_NOTENODAT)
| ELSE BAS\$SIGNAL_IO (BASSK_TOOLITDAT);

|+ Update the cursor position if this input was terminated by an escape.
| Save cursor position if last PRINT terminator was a semi or comma.
| Use BUDDY_PTR 'cuz we want to use the PRINT data base for channel 0

```

: 1351      2635 2      T_CCB = .CCB [LUBSA_BUDDY_PTR];
: 1352      2636 2      T_CCB [LUBSL_PRINT_POS] = TIF .CCB [RABSW_STV0] EQL K_ESCAPE AND .T_CCB [LUBSV_FORM_CHAR] EQLU 1
: 1353      2637 2      THEN .CCB [RABSW_RSZ] + .T_CCB [LUBSL_PRINT_POS] + 1
: 1354      2638 2      ELSE 0;
: 1355      2639 2
: 1356      2640 2
: 1357      2641 2      !+ Set RECOUNT to the number of bytes read
: 1358      2642 2      ! If the file is a terminal format file, then RECOUNT has to be
: 1359      2643 2      ! adjusted for the carriage control terminator.
: 1360      2644 2
: 1361      2645 2
: 1362      2646 2      RECOUNT = .CCB [RABSW_RSZ] + (IF .CCB [LUBSV_TERM_FOR] THEN SELECTONFU .CCB [RABSW_STV0] OF
: 1363      2647 2      SET
: 1364      2648 2      [K_ESCAPE] : .CCB [RABSW_STV2];
: 1365      2649 2      [K_CR] : 2;
: 1366      2650 2      [OTHERWISE] : 0;
: 1367      2651 2      TES ELSE 0);
: 1368      2652 2
: 1369      2653 1      RETURN 1;
END;                                         ! End of BASS$REC_RSL1

```

				52 DD 00000 BASS\$REC_RSL1::		
4F	A1	AB	04	E0 00002	PUSHL R2	2521
38	FE	AB	05	E1 00007	BBS #4, -95(CC(B), 4\$	2582
			5B	DD 0000C	BBC #5, -2(CC(B), 3\$	
			01	FB 0000E	PUSHL CCB	2586
			50	DO 00015	CALLS #1, SVSSGET	
			52	D1 00018	MOVL R0, RMS_STATUS	
			07	12 0001F	CMPL RM\$_STATUS, #67153	2588
			00	FB 00021	BNEQ 1\$	
			52	E8 00028	CALLS #0, BASS\$SIGNAL_CTRLC	
			7E	D4 0002B	BLBS RM\$ STATUS, 2\$	
			01	FB 0002D	CLRL -(SP)	2594
			01	C3 00032	CALLS #1, GET_ERROR	
80	AB	0000V	28	AB 3C 00038	SUBL3 #1, 40(CC(B), -80(CC(B)	2612
		28	50	9E 0003C	MOVZWL 34(CC(B) R0	2613
			84	AB	MOVAB #40(CC(B)[R0], -76(CC(B)	
			22	BB40	BRB 5\$	2582
			1D	11 00042	BBS #4, -95(CC(B), 4\$	2624
0D	A1	AB	04	E0 00044	MOVZBL #BASS\$K_NOTE_NODAT, -(SP)	2626
		7E	006	8F 9A 00049	CALLS #1, BASS\$SIGNAL	
			01	FB 0004D	BRB 5\$	
			0B	11 00054	MOVZBL #BASS\$K_TOOL_ITDAT, -(SP)	2628
			00G	8F 9A 00056	CALLS #1, BASS\$SIGNAL_10	
			01	FB 0005A	MOVL -72(CC(B), T_CC(B	2635
			51	AB DO 00061	MOVZWL 12(CC(B), R2	2636
			52	AB 3C 00065	CMPW R2, #27	
			1B	52 B1 00069	BNEQ 6\$	
			11	12 0006C	BBC #2, -2(T_CC(B), 6\$	
OC	FE	A1	02	E1 0006E	MOVZWL 34(CC(B), R0	2637
		50	22	AB 3C 00073	ADDL2 -56(T_CC(B), R0	
		50	C8	A1 C0 00077	INCL R0	
			50	D6 0007B	BRB 7\$	
			02	11 0007D	CLRL R0	2636
			50	D4 0007F	6\$:	

C 7
16-Sep-1984 01:01:12 VAX-11 Blfs-32 v4.0-742
14-Sep-1984 11:56:35 [BASRTL.SRC]BASREC PRO.B32;1

Page 36
(13)

; Routine Size: 179 bytes, Routine Base: _BASS\$CODE + 032C

: 1370 2654 1

```

: 1372      2655 1 GLOBAL ROUTINE BASSREC_MIN1           ! MAT Input element transmitter
: 1373      2656 1 : JSB_REC1 =
: 1374      2657 1
: 1375      2658 1 ++
: 1376      2659 1 FUNCTIONAL DESCRIPTION:
: 1377      2660 1
: 1378      2661 1 BASSREC_MIN1 reads one record and checks for a continuation character.
: 1379      2662 1 Then return start and end+1 of user
: 1380      2663 1 part of record to be processed as input.
: 1381      2664 1
: 1382      2665 1 FORMAL PARAMETERS:
: 1383      2666 1
: 1384      2667 1     NONE
: 1385      2668 1
: 1386      2669 1     IMPLICIT INPUTS:
: 1387      2670 1
: 1388      2671 1     LUBSW_RBUF_SIZE      Size of record buffer allocated in OPEN.
: 1389      2672 1     LUBSA_RBUF_ADR      Address of record buffer from OPEN.
: 1390      2673 1     LUBSV_TERM_DEV      flag indicating a terminal device.
: 1391      2674 1     RABSL_RBF        Pointer to buffer
: 1392      2675 1     RABSW_RSZ        buffer size
: 1393      2676 1
: 1394      2677 1     IMPLICIT OUTPUTS:
: 1395      2678 1
: 1396      2679 1     RECOUNT          Own storage for RECOUNT function.
: 1397      2680 1     LUBSA_BUF_PTR      points to first char of user part of
: 1398      2681 1     LUBSA_BUF_END       record buffer.
: 1399      2682 1     points to end+1 of user part of
: 1400      2683 1     record buffer.
: 1401      2684 1
: 1402      2685 1     ROUTINE VALUE:
: 1403      2686 1
: 1404      2687 1     NONE
: 1405      2688 1
: 1406      2689 1     SIDE EFFECTS:
: 1407      2690 1
: 1408      2691 1     Reads next record from file on this logical unit.
: 1409      2692 1     SIGNALS any resultant RMS errors.
: 1410      2693 1     --
: 1411      2694 1     BEGIN
: 1412      2695 2
: 1413      2696 2
: 1414      2697 2     EXTERNAL REGISTER
: 1415      2698 2     CCB : REF BLOCK [, BYTE];
: 1416      2699 2
: 1417      2700 2
: 1418      2701 2     LITERAL
: 1419      2702 2     K_ESCAPE = %X'1B',
: 1420      2703 2     K_CR = %X'0D';
: 1421      2704 2
: 1422      2705 2     LOCAL
: 1423      2706 2     RMS_STATUS,
: 1424      2707 2     T [CB : REF BLOCK [, BYTE],
: 1425      2708 2     STATUS;
: 1426      2709 2
: 1427      2710 2
: 1428      2711 2     ! Return status to UDF of whether
:                      ! to keep reading
:                      ++

```

```
1429      2712 2    ; Read record into buffer using RMS and check for errors and a continuation character
1430      2713 2    ; Signal any RMS errors directly.
1431      2714 2
1432      2715 2
1433      2716 2
1434      2717 2
1435      2718 2
1436      2719 2
1437      2720 2
1438      2721 2
1439      2722 2
1440      2723 2
1441      2724 2
1442      2725 2
1443      2726 2
1444      2727 2
1445      2728 2
1446      2729 2
1447      2730 2
1448      2731 2
1449      2732 2
1450      2733 2
1451      2734 2
1452      2735 2
1453      2736 2
1454      2737 2
1455      2738 2
1456      2739 2
1457      2740 2
1458      2741 2
1459      2742 2
1460      2743 2
1461      2744 2
1462      2745 2
1463      2746 2
1464      2747 2
1465      2748 2
1466      2749 2
1467      2750 2
1468      2751 2
1469      2752 2
1470      2753 2
1471      2754 2
1472      2755 3
1473      2756 3
1474      2757 4
1475      2758 4
1476      2759 4
1477      2760 4
1478      2761 3
1479      2762 3
1480      2763 3
1481      2764 3
1482      2765 3
1483      2766 3
1484      2767 3
1485      2768 3

; Read record into buffer using RMS and check for errors and a continuation character
; Signal any RMS errors directly.

IF .CCB [ISBSV_MAT_CONT]
THEN
  BEGIN
    RMS_STATUS = $GET (RAB = .CCB);
    IF .RMS_STATUS EQL RMSS_CONTROLC
    THEN
      BAS$SIGNAL_CTRLC ();
    IF NOT .RMS_STATUS
    THEN
      GET_ERROR (K_STOP);

!+
; Return start-1 and end+1 address of record just read
; LUBSA_BUF_PTR is set to the beginning-1 of the buffer only for BASIC
; input. This is seen as a solution to the problem of the user entering
; <return> as the response to a prompt (null input record) and an empty
; or depleted buffer which requires another Get.
; The algorithm:
; 1) Does LUBSA_BUF_PTR = LUBSA_BUF_END?
;    T: The buffer is depleted = another Get is required.
; 2) Add one to LUBSA_BUF_PTR
; 3) Does LUBSA_BUF_PTR = LUBSA_BUF_END?
;    T: Return the default value.
; 4) Scan for the next field.

!-
; CCB [LUBSA_BUF_PTR] = .CCB [RABSL_RBF] - 1;
; CCB [LUBSA_BUF_END] = .CCB [RABSL_RBF] + .CCB [RABSW_RSZ];

!+
; Check for an '8' as the last character of the record. If it is there,
; it is a continuation character and signifies that there is more data to
; come in the next record.
;-

IF .(.CCB [LUBSA_BUF_END] - 1)<0, 8> EQLU K_MAT_CONT_CHAR
THEN
  BEGIN
    CCB [LUBSA_BUF_END] = .CCB [LUBSA_BUF_END] - 1;
    CCB [ISBSV_MAT_CONT] = 1;
  END
ELSE
  CCB [ISBSV_MAT_CONT] = 0;

!+
; Update the cursor position if this input was terminated by an escape.
; Save the cursor position if last PRINT terminator was a semi or comma.
; Use BUDDY_PTR 'cuz we want to use the PRINT data base for channel 0
;-
```

```

1486    2769 3      T_CCB = .CCB [LUBSA_BUDDY_PTR];
1487    2770 4      T_CCB [LUBSL_PRINT_POS] = -(IF .CCB [RABSW_STVO] EQL K_ESCAPE AND .T_CCB [LUBSV_FORM_CHAR] EQLU 1
1488    2771 4      THEN .CCB [RABSW_RSZ] + .T_CCB [LUBSL_PRINT_POS] + T
1489    2772 4      ELSE 0);
1490    2773 3
1491    2774 3
1492    2775 3      |+
1493    2776 3      | Set RECOUNT to the number of bytes read
1494    2777 3      | If the file is a terminal format file, then RECOUNT has to be
1495    2778 3      | adjusted for the carriage control terminator.
1496    2779 3
1497    2780 4      RECOUNT = .CCB [RABSW_RSZ] + (IF .CCB [LUBSV_TERM_FOR] THEN SELECTONEU .CCB [RABSW_STVO] OF
1498    2781 4      SET
1499    2782 4      [K_ESCAPE] : .CCB [RABSW_STV2];
1500    2783 4      [K_CR] : 2;
1501    2784 4      [OTHERWISE] : 0;
1502    2785 4      TES ELSE 0);
1503    2786 3      STATUS = 1;
1504    2787 3      END
1505    2788 2      ELSE
1506    2789 2      STATUS = 0;
1507    2790 2
1508    2791 2      RETURN .STATUS;
1509    2792 1      END;

```

! End of BASSREC_MIN1

			S2	DD 00000 BASSREC_MIN1::			
03	97 AB		01	E0 00002	PUSHL	R2	2655
			009E	31 00007	BBS	#1 -105(CCB), 18	2716
			58	DD 0000A	1\$: PUSHL	CCB	2720
	00000000G 00		01	FB 0000C	CALLS	#1. SYSSGET	2722
	52		50	D0 00013	MOVL	R0, RMS STATUS	2724
	00010651 8F		52	D1 00016	CMPL	RMS_STATUS, #67153	2726
	00000000G 00		07	12 0001D	BNEQ	28	2728
	07		00	FB 0001F	CALLS	#0, BASSSIGAL_CTRLC	2746
	52		52	E8 00026	BLBS	RMS STATUS, 38	2747
	00000000V CF		7E	D4 00029	CLRL	-(SP)	2755
	28 AB		01	FB 0002B	CALLS	#1. GET ERROR	2758
	50		01	C5 00030	3\$: SUBL3	#1. 40(CCB), -80(CCB)	2759
	B4 AB		22	AB 3C 00036	MOVZWL	34(CCB), R0	2762
	50		28	BB40 9E 0003A	MOVAB	240(CCB)[R0], -76(CCB)	2769
	26		84	AB D0 00040	MOVL	-76(CCB), R0	2770
	FF		A0	91 00044	CMPB	-1(R0), #38	
			09	12 00048	BNEQ	48	
			B4	AB D7 0004A	DECL	-76(CCB)	
	97 AB		02	88 0004D	BISB2	#2. -105(CCB)	
			04	11 00051	BRB	58	
	97 AB		02	8A 00053	4\$: BICB2	#2. -105(CCB)	
	50		AB D0 00057	5\$: MOV	-72(CCB), T_CCB		
	52		88 OC	AB 3C 0005B	MOVZWL	12(CCB), R2-	
	18		52	B1 0005F	CMPW	R2, #27	
			11	12 00062	BNEQ	68	
	OC FE A0		02	E1 00064	BBC	#2. -2(T_CCB), 68	

		51	22	AB 3C 00069	MOVZWL 34(CCB), R1	2771
		A0 C0 0006D		ADDL2 -56(T_CCB), R1		
		51 D6 00071		INCL R1		
		02 11 00073		BRB 7S		
		51 D4 00075	6\$:	CLRL R1	2770	
		51 D0 00077	7\$:	MOVL R1. -56(T_CCB)		
		04 E1 0007B		BBC #4. -2(CCB), 9S	2780	
15	C8 FE	A0 AB 1B		CMPW R2, #27	2782	
		06 12 00083		BNEQ 8S		
		50 0E AB 3C 00085		MOVZWL 14(CCB), R0		
		0C 11 00089		BRB 10S		
		0D 52 B1 0008B	8\$:	CMPW R2, #13	2783	
		05 12 0008E		BNEQ 9S		
		50 02 D0 00090		MOVL #2 R0		
		02 11 00093		BRB 10S		
		50 51 22 AB 3C 00097	9\$: 10\$:	CLRL R0	2780	
00000000' EF		50 01 D0 000A3		MOVZWL 34(CCB), R1		
		02 11 000A6		ADDL3 R1, R0, RECOUNT	2786	
		50 D4 000AB	11\$:	MOVL #1 STATUS	2716	
		04 BA 000AA	12\$:	BRB 12S	2789	
		05 000AC		CLRL STATUS		
				POPR #^M<R2>	2792	
				RSB		

; Routine Size: 173 bytes. Routine Base: _BASS\$CODE + 03DF

; 1510 2793 1

: 1512 2794 1 GLOBAL ROUTINE BASS\$REC_RSL9 ! Read IO_END
.: 1513 2795 1 : JSB_REC9 NOVALUE !
.: 1514 2796 1
.: 1515 2797 1 ++
.: 1516 2798 1 FUNCTIONAL DESCRIPTION:
.: 1517 2799 1 BASS\$REC_RSL9 is a no-op!
.: 1518 2800 1
.: 1519 2801 1 FORMAL PARAMETERS:
.: 1520 2802 1 NONE
.: 1521 2803 1
.: 1522 2804 1
.: 1523 2805 1
.: 1524 2806 1 IMPLICIT INPUTS:
.: 1525 2807 1 NONE
.: 1526 2808 1
.: 1527 2809 1
.: 1528 2810 1 IMPLICIT OUTPUTS:
.: 1529 2811 1
.: 1530 2812 1 ROUTINE VALUE:
.: 1531 2813 1 NONE
.: 1532 2814 1
.: 1533 2815 1
.: 1534 2816 1 SIDE EFFECTS:
.: 1535 2817 1
.: 1536 2818 1
.: 1537 2819 1
.: 1538 2820 2 BEGIN
.: 1539 2821 2 RETURN;
.: 1540 2822 1 END;
 ! End of BASS\$REC_RSL9

05 00000 BASS\$REC_RSL9::
RSB

: 2822

: Routine Size: 1 bytes, Routine Base: _BASSCODE + 048C

: 1541 2823 1

: 1543 2824 1 GLOBAL ROUTINE BASS\$REC_MIN9
: 1544 2825 1 : JSB_REC9 NOVALUE :=
: 1545 2826 1
: 1546 2827 1
: 1547 2828 1 ++
: 1548 2829 1 FUNCTIONAL DESCRIPTION:
: 1549 2830 1 BASS\$REC_RSL9 is a no-op!
: 1550 2831 1
: 1551 2832 1 FORMAL PARAMETERS:
: 1552 2833 1 NONE
: 1553 2834 1
: 1554 2835 1
: 1555 2836 1
: 1556 2837 1
: 1557 2838 1
: 1558 2839 1
: 1559 2840 1
: 1560 2841 1
: 1561 2842 1 ROUTINE VALUE:
: 1562 2843 1 NONE
: 1563 2844 1
: 1564 2845 1
: 1565 2846 1
: 1566 2847 1
: 1567 2848 1
: 1568 2849 1
: 1569 2850 2 SIDE EFFECTS:
: 1570 2851 2 --
: 1571 2852 1 BEGIN
: RETURN;
: END;
: ! End of BASS\$REC_MIN9

05 00000 BASS\$REC_MIN9::
RSB

: 2852

; Routine Size: 1 bytes, Routine Base: _BASSCODE + 04BD

: 1572 2853 1

1574 2854 1 GLOBAL ROUTINE BASS\$REC_MLI1 : JSB_RECT = : MAT Linput element transmitter

1575 2855 1

1576 2856 1

1577 2857 1

1578 2858 1

1579 2859 1

1580 2860 1

1581 2861 1

1582 2862 1

1583 2863 1

1584 2864 1

1585 2865 1

1586 2866 1

1587 2867 1

1588 2868 1

1589 2869 1

1590 2870 1

1591 2871 1

1592 2872 1

1593 2873 1

1594 2874 1

1595 2875 1

1596 2876 1

1597 2877 1

1598 2878 1

1599 2879 1

1600 2880 1

1601 2881 1

1602 2882 1

1603 2883 1

1604 2884 1

1605 2885 1

1606 2886 1

1607 2887 1

1608 2888 1

1609 2889 1

1610 2890 1

1611 2891 1

1612 2892 1

1613 2893 1

1614 2894 1

1615 2895 2

1616 2896 2

1617 2897 2

1618 2898 2

1619 2899 2

1620 2900 2

1621 2901 2

1622 2902 2

1623 2903 2

1624 2904 2

1625 2905 2

1626 2906 2

1627 2907 2

1628 2908 2

1629 2909 2

1630 2910 2

++ FUNCTIONAL DESCRIPTION:
BASS\$REC_MLI1 unconditionally reads one record. There is no continuation character for MAT LINPUT. Otherwise an error is signalled. Then return start and end+1 of user part of record to be processed as input.

FORMAL PARAMETERS:
NONE

IMPLICIT INPUTS:
LUBSW_RBUF_SIZE Size of record buffer allocated in OPEN.
LUBSA_RBUF_ADDR Address of record buffer from OPEN.
RABSL_RBF Pointer to buffer
RABSW_RSZ buffer size

IMPLICIT OUTPUTS:
RECOUNT Own storage for RECOUNT function.
LUBSA_BUF_PTR points to first char of user part of record buffer.
LUBSA_BUF_END points to end+1 of user part of record buffer.

ROUTINE VALUE:
NONE

SIDE EFFECTS:
Reads next record from file on this logical unit.
SIGNS any resultant RMS errors.

--

BEGIN

EXTERNAL REGISTER
[CB : REF BLOCK [, BYTE];

LOCAL
RMS STATUS,
T_CB : REF BLOCK [, BYTE];

LITERAL
K_ESCAPE = %X'1B',
K_CR = %X'0D';

!+
! Read record into buffer using RMS and check for errors
! Signal any RMS errors directly.

```

1631      2911 2      !-
1632      2912 2
1633      2913 2
1634      2914 2
1635      2915 2
1636      2916 2
1637      2917 2
1638      2918 2
1639      2919 2
1640      2920 2
1641      2921 2
1642      2922 2
1643      2923 2
1644      2924 2      + Return start-1 and end+1 address of record just read
1645      2925 2      LUBSA_BUF_PTR is set to the beginning-1 of the buffer only for BASIC
1646      2926 2      Input. This is seen as a solution to the problem of the user entering
1647      2927 2      <return> as the response to a prompt (null input record) and an empty
1648      2928 2      or depleted buffer which requires another Get.
1649      2929 2      The algorithm:
1650      2930 2      1) Does LUBSA_BUF_PTR = LUBSA_BUF_END?
1651      2931 2      T: The buffer is depleted - another Get is required.
1652      2932 2      2) Add one to LUBSA_BUF_PTR
1653      2933 2      3) Does LUBSA_BUF_PTR = LUBSA_BUF_END?
1654      2934 2      T: Return the default value.
1655      2935 2      4) Scan for the next field.

1656      2936 2
1657      2937 2
1658      2938 2      CCB[LUBSA_BUF_PTR] = .CCB[RABSL_RBF] - 1;
1659      2939 2      CCB[LUBSA_BUF_END] = .CCB[RABSL_RBF] + .CCB[RABSW_RSZ];
1660      2940 2
1661      2941 2      + Update the cursor position if this input was terminated by an escape.
1662      2942 2      Save the cursor position if last PRINT terminator was a semi or comma.
1663      2943 2      Use BUDDY_PTR 'cuz we want to use the PRINT data base for channel 0
1664      2944 2
1665      2945 2      T_CCB = .CCB[LUBSA_BUDDY_PTR];
1666      2946 2      T_CCB[LUBSL_PRINT_POS] = (IF .CCB[RABSW_STV0] EQL K_ESCAPE AND .T_CCB[LUBSV_FORM_CHAR] EQLU 1
1667      2947 2          THEN .CCB[RABSW_RSZ] + .T_CCB[LUBSL_PRINT_POS] + 1
1668      2948 2          ELSE 0);
1669      2949 2
1670      2950 2      + Set RECOUNT to the number of bytes read
1671      2951 2      If the file is a terminal format file, then RECOUNT has to be
1672      2952 2      adjusted for the carriage control terminator.
1673      2953 2
1674      2954 2      RECOUNT = .CCB[RABSW_RSZ] + (IF .CCB[LUBSV_TERM_FOR] THEN SELECTONEU .CCB[RABSW_STV0] OF
1675      2955 2          SET
1676      2956 2              [K_ESCAPE] : .CCB[RABSW_STV2];
1677      2957 2              [K_CR] : 2;
1678      2958 2              [OTHERWISE] : 0;
1679      2959 2              TES ELSE 0);
1680      2960 2      RETURN 1
1681      2961 1      END;                                ! End of BASSREC_ML1
```

				PUSHL R2	2854
				PUSHL CCB	2913
				CALLS #1, SYS\$GET	
				MOVL R0, RMS STATUS	
				CMPBL RM\$_STATUS, #67153	
				BNEQ 1\$	2915
				CALLS #0, BAS\$SIGNAL_CTRLC	2917
				BLBS RM\$ STATUS, 2\$	2919
				CLRL -(SP)	2921
				CALLS #1, GET_ERROR	
				SUBL3 #1, 40(CCB), -80(CCB)	2938
				MOVZWL 34(CCB), R0	2939
				MOVAB 40(CCB)[R0], -76(CCB)	
				MOVL -72(CCB), T_CCB	2945
				MOVZWL 12(CCB), R2	2946
				CMPW R2, #27	
				BNEQ 3\$	
				BBC #2, -2(T_CCB), 3\$	
				MOVZWL 34(CCB), R0	2947
				ADDL2 -56(T_CCB), R0	
				INCL R0	
				BRB 4\$	
				CLRL R0	2946
				MOVL R0, -56(T_CCB)	
				BBC #4, -2(CCB), 6\$	2954
				CMPW R2, #27	2956
				BNEQ 5\$	
				MOVZWL 14(CCB), R0	
				BRB 7\$	
				CMPW R2, #13	2957
				BNEQ 6\$	
				MOVL #2, R0	
				BRB 7\$	
				CLRL R0	2954
				MOVZWL 34(CCB), R1	
				ADDL3 R1, R0, RECOUNT	2960
				MOVL #1, R0	2961
				POPR #^H<R2>	
				RSB	

: Routine Size: 138 bytes. Routine Base: _BASSCODE + 048E

: 1682 2962 1

```

1684 2963 1 GLOBAL ROUTINE BASSREC_WSL0           ! Write list-directed
1685 2964 1 : JSB_REC0 NOVALUE =
1686 2965 1
1687 2966 1
1688 2967 1 ++
1689 2968 1 FUNCTIONAL DESCRIPTION:
1690 2969 1 BASSREC_WSL0 prepares a record for list-directed output.
1691 2970 1 Then return start and end+1 of user
1692 2971 1 part of record to be processed.
1693 2972 1
1694 2973 1 FORMAL PARAMETERS:
1695 2974 1
1696 2975 1
1697 2976 1
1698 2977 1
1699 2978 1
1700 2979 1
1701 2980 1
1702 2981 1
1703 2982 1
1704 2983 1
1705 2984 1
1706 2985 1
1707 2986 1
1708 2987 1
1709 2988 1
1710 2989 1
1711 2990 1
1712 2991 1
1713 2992 1
1714 2993 1
1715 2994 1
1716 2995 1
1717 2996 1
1718 2997 1
1719 2998 1
1720 2999 1
1721 3000 1
1722 3001 1
1723 3002 1
1724 3003 2
1725 3004 2
1726 3005 2
1727 3006 2
1728 3007 2
1729 3008 2
1730 3009 2
1731 3010 2
1732 3011 2
1733 3012 2
1734 3013 2
1735 3014 2
1736 3015 2
1737 3016 2
1738 3017 2
1739 3018 2
1740 3019 2

GLOBAL ROUTINE BASSREC_WSL0           ! Write list-directed
: JSB_REC0 NOVALUE =
+++
FUNCTIONAL DESCRIPTION:
BASSREC_WSL0 prepares a record for list-directed output.
Then return start and end+1 of user
part of record to be processed.

FORMAL PARAMETERS:
NONE

IMPLICIT INPUTS:
LUBSW_RBUF_SIZE      Size (bytes) allocated for record buffer at OPEN.
LUBSA_RBUF_ADR        Address of record buffer allocated at OPEN
LUBSV_FIXED           1 if fixed-length records
LUBSV_FORM_CHAR       Indicates that the last element transmitter ended
                      in a comma or semicolon format char.
LUBSV_FORCEABLE       Indicates a forcible device
LUBSV_CCO             Cancel control 0

IMPLICIT OUTPUTS:
LUBSB_BAS_VFC1        'Pre' carriage control
LUBSB_BAS_VFC2        'Post' carriage control
LUBSA_BUF_PTR          pointer to next byte of buffer
LUBSA_BUF_END          pointer to byte following the buffer
RABSV_CCO              Cancel control 0

ROUTINE VALUE:
NONE

SIDE EFFECTS:
--+
BEGIN
EXTERNAL REGISTER
  CCB : REF BLOCK [. BYTE];
+-
Copy the current status of the cancel-control-0 bit in the LUB
(possibly set by RCTRL0) into the RAB, and clear it from the
LUB. The net effect of this is that if the bit is set in the
LUB, then the CANTRL0 modifier will be applied to this write
operation only.
-
CCB [RABSV_CCO] = CCB [LUBSV_CCO];
CCB [LUBSV_CCO] = 0;
+-

```

```

1741      3020  2 | If there is a format character pending and this is not a forcible
1742      3021  2 | device, then don't change the buffer pointers. The PUT will be done when
1743      3022  2 |
1744      3023  2 |
1745      3024  2 |
1746      3025  2 |
1747      3026  2 |
1748      3027  2 |
1749      3028  2 |
1750      3029  2 |
1751      3030  2 |
1752      3031  2 |
1753      3032  2 |
1754      3033  2 | CCB [LUB$B_BAS_VFC1] = (IF .CCB [LUB$V_FORM_CHAR] THEN BASSK_NULL ELSE BASSK_LF);
1755      3034  2 | CCB [LUB$B_BAS_VFC2] = BASSK_NULL;
1756      3035  2 |
1757      3036  2 |
1758      3037  2 |
1759      3038  2 |
1760      3039  2 |
1761      3040  2 |
1762      3041  2 |
1763      3042  2 |
1764      3043  2 | IF NOT .CCB [LUB$V_OUTBUF_DR]
1765      3044  3 | THEN
1766      3045  3 | BEGIN
1767      3046  3 |   CCB [LUB$A_BUF_PTR] = .CCB [LUB$A_RBUF_ADR];
1768      3047  3 |   CCB [LUB$A_BUF_END] = .CCB [LUB$A_RBUF_ADR] + .CCB [LUB$W_RBUF_SIZE];
1769      3048  3 | END;
1770      3049  2 |
1771      3050  1 | RETURN;
END;                                         ! END OF BASS$REC_WSL0

```

	50	A0	AB	01	02	EF 00000 BASS\$REC_WSL0::			
07	AB	01		07	50	F0 00006	EXTZV	#2, #1, -96(CCB), R0	3016
		A0	AB		04	8A 0000C	INSV	R0, #7, #1, 7(CCB)	3017
		FE	AB		02	E1 00010	BICB2	#4, -96(CCB)	3025
		24	AB		06	E1 00015	BBC	#2, -2(CCB), 1\$	
		04	FE	AB	02	E1 0001A	BBC	#6, -2(CCB), 3\$	3033
					50	D4 0001F	CLRL	#2, -2(CCB), 1\$	
					03	11 00021	BRB	2\$	
					01	D0 00023	1\$:	MOV L	
					50	9B 00026	2\$:	MOV ZBW	3042
		OF	DA	AB	03	E0 0002A	BBS	R0, -38(CCB)	3045
			FE	AB	00	D0 0002F	MOVL	#3, -2(CCB), 3\$	
			BO	AB	50	D2 AB	MOVZWL	-20(CCB), -80(CCB)	3046
					EC	BB40	MOVAB	-46(CCB), R0	
					05	9E 00038	RSB	2-20(CCB)[R0], -76(CCB)	3050
					05	0003E	3\$:		

; Routine Size: 63 bytes, Routine Base: _BASS\$CODE + 0518

; 1772 3051 1

```

: 1774      3052 1 GLOBAL ROUTINE BASS$REC_WSL1 (           ! Write list-directed
: 1775      3053 1   CARRIAGE_CTRL) : JSB_REC_WSL1 NOVALUE =           ! Called from BASS$DO_WRITE
: 1776      3054 1
: 1777      3055 1 ++
: 1778      3056 1   FUNCTIONAL DESCRIPTION:
: 1779      3057 1
: 1780      3058 1   Write one list-directed record and initialize for the next
: 1781      3059 1   BASS$REC_WSL1 writes one output buffer and then
: 1782      3060 1   initializes the output buffer and returns start and end+1 of user
: 1783      3061 1   part of record buffer to be filled by caller.
: 1784      3062 1   If this routine is called because the buffer overflowed then the 'post'
: 1785      3063 1   carriage control should be null. If this routine is called because the
: 1786      3064 1   margin overflowed, then the 'post' carriage control should be 'CR'.
: 1787      3065 1
: 1788      3066 1   FORMAL PARAMETERS:
: 1789      3067 1
: 1790      3068 1   CARRIAGE_CTRL.rlu.v    carriage control for the record
: 1791      3069 1
: 1792      3070 1   IMPLICIT INPUTS:
: 1793      3071 1
: 1794      3072 1   LUBSW_RBUF_SIZE      Size (bytes) allocated for record buffer at OPEN.
: 1795      3073 1   LUBSA_RBUF_ADR       Address of record buffer allocated at OPEN
: 1796      3074 1   LUBSA_BUF_PTR        Pointer to next byte in user buffer.
: 1797      3075 1   RABSL_RBF             Pointer to user buffer
: 1798      3076 1
: 1799      3077 1   IMPLICIT OUTPUTS:
: 1800      3078 1
: 1801      3079 1   LUBSB_BAS_VFC1      'Pre' carriage control
: 1802      3080 1   LUBSB_BAS_VFC2      'Post' carriage control
: 1803      3081 1   LUBSA_BUF_PTR        Pointer to start of user part of record buffer
: 1804      3082 1   LUBSA_BUF_END        Pointer to end+1 of user part of record buffer
: 1805      3083 1   LUBSV_OUTBUF_DR     Indicates valid data in the output buffer
: 1806      3084 1   RABSW_RSZ            Size of user buffer
: 1807      3085 1
: 1808      3086 1   ROUTINE VALUE:
: 1809      3087 1
: 1810      3088 1   NONE
: 1811      3089 1
: 1812      3090 1   SIDE EFFECTS:
: 1813      3091 1
: 1814      3092 1   Writes one RMS sequential record.
: 1815      3093 1   SIGNALS BASSK_FATSYSIO on PUT error.
: 1816      3094 1
: 1817      3095 1
: 1818      3096 2   BEGIN
: 1819      3097 2
: 1820      3098 2   EXTERNAL REGISTER
: 1821      3099 2   CCB = 11 : REF BLOCK [. BYTE];
: 1822      3100 2
: 1823      3101 2   LITERAL
: 1824      3102 2   K_NO_CR = 2;
: 1825      3103 2
: 1826      3104 2   LOCAL
: 1827      3105 2   RMS_STATUS;
: 1828      3106 2
: 1829      3107 2
: 1830      3108 2   ! Set 'post' carriage control to CR or NULL depending on whether the margin

```

```

: 1831      3109 2 ! overflowed or the buffer overflowed.
: 1832      3110 2 ! if this is a file, the carriage control is ignored and a record is PUT.
: 1833      3111 2
: 1834      3112 2 CCB [LUBSB_BAS_VFC2] = (IF .CARRIAGE_CTRL EQL BASSK_BUF_EXC THEN BASSK_NULL ELSE BASSK_(R));
: 1835      3113 2
: 1836      3114 2
: 1837      3115 2 ! perform the record write.
: 1838      3116 2 Set recordsize to actual length of record
: 1839      3117 2
: 1840      3118 2
: 1841      3119 2 CCB [RABSW_RSZ] = .CCB [LUBSA_BUF_PTR] - .CCB [LUBSA_BUF_BEG];
: 1842      3120 2
: 1843      3121 2 ! Output buffer to RMS and check for errors
: 1844      3122 2 If errors, SIGNAL_STO
: 1845      3123 2
: 1846      3124 2
: 1847      3125 2
: 1848      3126 2 CCB [RABSL_RBF] = .CCB [LUBSA_BUF_BEG];
: 1849      3127 2 CCB [LUBSV_OUTBUF_DR] = 0;
: 1850      3128 2 RMS_STATUS = SPUT (RAB = .CCB);
: 1851      3129 2
: 1852      3130 2 IF .RMS_STATUS EQL RMSS_CONTROLC
: 1853      3131 2 THEN
: 1854      3132 2     BASS$SIGNAL_CTRLC ();
: 1855      3133 2
: 1856      3134 2
: 1857      3135 2
: 1858      3136 2
: 1859      3137 2
: 1860      3138 2
: 1861      3139 2
: 1862      3140 2 ! Set the 'pre' carriage control to LF if CARRIAGE_CTRL warrants it.
: 1863      3141 2 ! Set the 'post' carriage control to null.
: 1864      3142 2
: 1865      3143 2
: 1866      3144 2 CCB [LUBSB_BAS_VFC1] = (IF .CARRIAGE_CTRL EQL BASSK_BUF_EXC THEN BASSK_NULL ELSE BASSK_LF);
: 1867      3145 2 CCB [LUBSB_BAS_VFC2] = BASSK_NULL;
: 1868      3146 2
: 1869      3147 2 ! Initialize record buffer for another list-directed write
: 1870      3148 2 return record buffer pointers to caller
: 1871      3149 2
: 1872      3150 2
: 1873      3151 2
: 1874      3152 2 CCB [LUBSA_BUF_PTR] = .CCB [LUBSA_RBUF_ADR];
: 1875      3153 2 CCB [LUBSA_BUF_END] = .CCB [LUBSA_RBUF_ADR] + .CCB [LUBSW_RBUF_SIZE];
: 1876      3154 2 RETURN;
: 1877      3155 1 END;                                ! End of routine - BASS$UDF_WSL1

```

OC BB 00000 BASSREC_WSL1::
 08 53 D4 00002 PUSH R4 #^MCR2,R3>
 50 D1 00004 CLR L R3
 06 12 00007 CMPL CARRIAGE_CTRL, #8
 BNEQ 1S

3052
3112

			53	D6 00009	INCL R3	
			50	D4 00008	CLRL R0	
			04	11 0000D	BRB 2\$	
22 AB	DB AB	BD	8F	9A 0000F	MOVZBL #141, R0	3119
	BO AB	BC	50	90 00013	MOVB R0 -37(CCB)	3126
	28 AB	BC	AB	A3 00017	SUBW3 -68(CCB), -80(CCB), 34(CCB)	3127
	FE AB	BC	AB	DD 0001E	MOVL -68(CCB), 40(CCB)	3129
			08	8A 00023	BICB2 #8, -2(CCB)	
			5B	DD 00027	PUSHL CCB	
	00000000G 00		01	FB 00029	CALLS #1, SYSSPUT	
	52		50	DD 00030	MOVL R0, RMS STATUS	
	00010651 8F		52	D1 00033	CMPL RM\$ STATUS, #67153	3131
	00000000G 00		07	12 0003A	BNEQ 3S	
	07		00	FB 0003C	CALLS #0, BASS\$SIGNAL_CTRLC	3133
	0000V CF		52	E8 00043	BLBS RMS STATUS, 4S	3135
	04		7E	D4 00046	CLRL -(SP)	3137
			01	FB 00048	CALLS #1, PUT_ERROR	
			53	E9 00040	BLBC R3, 5S	3144
			50	D4 00050	CLRL R0	
			03	11 00052	BRB 6S	
	DA AB		01	DD 00054	MOVL #1, R0	
	BO AB		50	9B 00057	MOVZBW R0, -38(CCB)	3152
	50	EC AB	AB	DD 00058	MOVL -20(CCB), -80(CCB)	3153
	B4 AB	D2 EC BB40	3C	00060	MOVZWL -46(CCB), R0	3155
		OC	9E	00064	MOVAB 2-20(CCB)[R0], -76(CCB)	
			0C	BA 0006A	POPR #^M<R2,R3>	
			05	0006C	RSB	3155

: Routine Size: 109 bytes. Routine Base: _BASS\$CODE + 0557

: 1878 3156 1

1880 3157 1 GLOBAL ROUTINE BASS\$REC_RMFO
1881 3158 1 : JSB_REC0 NOVALUE = ! Initialize read memory formatted
1882 3159 1
1883 3160 1 ++
1884 3161 1 FUNCTIONAL DESCRIPTION:
1885 3162 1
1886 3163 1 Pick up pointer to last major frame from ISB and initialize BUF_BEG,
1887 3164 1 BUF_PTR, and BUF_END to the values for the data area found in the
1888 3165 1 frame.
1889 3166 1
1890 3167 1 FORMAL PARAMETERS:
1891 3168 1
1892 3169 1 NONE
1893 3170 1
1894 3171 1 IMPLICIT INPUTS:
1895 3172 1
1896 3173 1 ISBSA_MAJ_F_PTR pointer to last Basic major frame
1897 3174 1
1898 3175 1 IMPLICIT OUTPUTS:
1899 3176 1
1900 3177 1 ROUTINE VALUE:
1901 3178 1
1902 3179 1 NONE
1903 3180 1
1904 3181 1 SIDE EFFECTS:
1905 3182 1
1906 3183 1 NONE
1907 3184 1
1908 3185 1
1909 3186 1
1910 3187 2 --
1911 3188 2 BEGIN
1912 3189 2
1913 3190 2 EXTERNAL REGISTER
1914 3191 2 CCB = K_CCB_REG : REF_BLOCK [, BYTE];
1915 3192 2 LOCAL
1916 3193 2 BMF : REF_BLOCK [0, BYTE] FIELD (BSF\$MAJOR_FRAME);
1917 3194 2
1918 3195 2
1919 3196 2
1920 3197 2
1921 3198 2
1922 3199 2
1923 3200 2
1924 3201 2
1925 3202 2
1926 3203 2
1927 3204 2
1928 3205 2
1929 3206 2
1930 3207 2
1931 3208 2
1932 3209 2
1933 3210 2
1934 3211 2
1935 3212 2
1936 3213 2
1+ Reach back into the last major frame by picking up the value of R11 stored
in the ISB. Initialize BUF_PTR, BUF_END, BUF_BEG so that this will look
like a vanilla INPUT.
1-
BMF = .CCB [ISBSA_MAJ_F_PTR];
1+ If this cell is zero, then there was no DATA statement and an error should be
signalled.
1-
IF .BMF [BSFSA_CUR_DTA] EQLA 0 THEN BASS\$STOP_IO (BASS\$K_OUTOF_DAT);
CCB [LUBSA_BUF_BEG] = .BMF [BSFSA_CUR_DTA];
CCB [LUBSA_BUF_END] = .BMF [BSFSA_END_DTA];
1+ Subtract one from CUR_DATA for INPUT element transmitter compatibility.

```
: 1937      3214  2      !-
: 1938      3215  2
: 1939      3216  2      CCB [LUBSA_BUF_PTR] = .BMF [BSFSA_CUR_DTA] - 1;
: 1940      3217  2      RETURN;
: 1941      3218  1      END;
```

			DC	BB 00000 BASS\$REC RMFO::				
				PUSHR	#^M<R2,R3>			
	52	FF48	CB	D0 00002	MOVL	-184(CCB), BMF	3157	
	53	0087	C2	D0 00007	MOVL	135(BMF), R3	3201	
			0B	12 0000C	BNEQ	18	3207	
00000000G	7E	006	8F	9A 0000E	MOVZBL	#BASS\$K OUTOF DAT, -(SP)		
	00		01	FB 00012	CALLS	#1, BASS\$STOP_10		
	BC	AB	53	D0 00019	18:	MOVL	R3, -68(CCB)	3209
	B4	AB	008B	C2 0001D	MOVL	139(BMF), -76(CCB)	3210	
	B0	AB	FF	A3 9E 00023	MOVAB	-1(R3), -80(CCB)	3216	
			0C	BA 00028	POPR	#^M<R2,R3>	3218	
			05	0002A	RSB			

: Routine Size: 43 bytes. Routine Base: _BASS\$CODE + 05C4

: 1942 3219 1

```

1944      3220 1 GLOBAL ROUTINE BASS$REC_MRE1           ! Mat Read element transmitter
1945      3221 1 : JSB_REC1 =
1946      3222 1 ++
1947      3223 1 FUNCTIONAL DESCRIPTION:
1948      3224 1 Since MAT READ just takes as much input data as it can get, it will just
1949      3225 1 return a failure here because there is no more data.
1950      3226 1
1951      3227 1 FORMAL PARAMETERS:
1952      3228 1
1953      3229 1
1954      3230 1
1955      3231 1
1956      3232 1
1957      3233 1
1958      3234 1
1959      3235 1
1960      3236 1
1961      3237 1
1962      3238 1
1963      3239 1
1964      3240 1
1965      3241 1
1966      3242 1
1967      3243 1
1968      3244 1
1969      3245 1
1970      3246 1
1971      3247 1
1972      3248 1
1973      3249 1
1974      3250 1
1975      3251 1
1976      3252 2
1977      3253 2
1978      3254 1 -- BEGIN
                           RETURN 0
                           END;                                ! end of BASS$REC_MRE1

```

50 D4 00000 BASS\$REC_MRE1::
 05 00002 CLR R0
 RSB

; Routine Size: 3 bytes, Routine Base: _BASSCODE + 05EF

; 1979 3255 1

: 3253
: 3254

```

1981      3256 1 GLOBAL ROUTINE BASS$REC_RMF1           ! Read element transmitter
1982      3257 1 : JSB_REC1 NOVALUE =
1983      3258 1
1984      3259 1 ++
1985      3260 1 FUNCTIONAL DESCRIPTION:
1986      3261 1
1987      3262 1 BASS$REC_RMF1 should not be called in normal processing and will signal
1988      3263 1 an error (BASSK_OUTOF_DAT) if it is called.
1989      3264 1
1990      3265 1 FORMAL PARAMETERS:
1991      3266 1
1992      3267 1     NONE
1993      3268 1
1994      3269 1 IMPLICIT INPUTS:
1995      3270 1
1996      3271 1     NONE
1997      3272 1
1998      3273 1 IMPLICIT OUTPUTS:
1999      3274 1
2000      3275 1 ROUTINE VALUE:
2001      3276 1
2002      3277 1     NONE
2003      3278 1
2004      3279 1 SIDE EFFECTS:
2005      3280 1
2006      3281 1     Signal - BASSK_OUTOF_DAT - Out of Data
2007      3282 1
2008      3283 1 --
2009      3284 1
2010      3285 2 BEGIN
2011      3286 2     BASS$SIGNAL (BASSK_OUTOF_DAT);
2012      3287 2     RETURN;
2013      3288 1     END;                                ! end of BASS$REC_RMF1

```

	7E	00G	8F	9A	00000 BASS\$REC_RMF1::	
00000000G	00		01	FB	00004	MOVZBL #BASSK_OUTOF_DAT, -(SP)
			05	0000B		CALLS #1, BASS\$SIGNAL
						RSB

; Routine Size: 12 bytes, Routine Base: _BASSCODE + 05F2

; 2014 3289 1

: 3286
: 3288

```

: 2016      3290 1 GLOBAL ROUTINE BASS$REC_RMF9           ! Read IO_END
: 2017      3291 1 : JSB_REC9 NOVALUE =
: 2018      3292 1
: 2019      3293 1
: 2020      3294 1 ++
: 2021      3295 1 FUNCTIONAL DESCRIPTION:
: 2022      3296 1 Update the current data pointer in the last Basic major frame
: 2023      3297 1
: 2024      3298 1 FORMAL PARAMETERS:
: 2025      3299 1     NONE
: 2026      3300 1
: 2027      3301 1 IMPLICIT INPUTS:
: 2028      3302 1     NONE
: 2029      3303 1
: 2030      3304 1 IMPLICIT OUTPUTS:
: 2031      3305 1
: 2032      3306 1 ROUTINE VALUE:
: 2033      3307 1     NONE
: 2034      3308 1
: 2035      3309 1 SIDE EFFECTS:
: 2036      3310 1
: 2037      3311 1
: 2038      3312 1
: 2039      3313 1
: 2040      3314 1
: 2041      3315 1
: 2042      3316 2 --
: 2043      3317 2 BEGIN
: 2044      3318 2 EXTERNAL REGISTER
: 2045      3319 2     CCB = K_CCB_REG : REF_BLOCK [0, BYTE];
: 2046      3320 2
: 2047      3321 2 LOCAL
: 2048      3322 2     BMF : REF_BLOCK [0, BYTE] FIELD (BSF$MAJOR_FRAME);
: 2049      3323 2
: 2050      3324 2
: 2051      3325 2
: 2052      3326 2
: 2053      3327 2 ++
: 2054      3328 2     Set the current data pointer in the frame to BUF_PTR + 1.
: 2055      3329 2     The one is added because Input initialize will subtract one from BUF_PTR.
: 2056      3330 2     This whole matter is explained in IO_BEG.
: 2057      3331 2
: 2058      3332 2
: 2059      3333 2     BMF = .CCB [ISBSA_MAJ_F_PTR];
:                  BMF [BSF$A_CUR_DTA] = .CCB [LUBSA_BUF_PTR] + 1;
:                  RETURN;
:                  END;                                ! End of routine BASS$REC_RMF9

```

		50	FF48	CB	DD 00000 BASS\$REC_RMF9::	
0087	C0	B0	AB	01	C1 00005	MOV _L -184(CC _B), BMF
				05	0000C	ADDL3 #1, -80(CC _B), 135(BMF)
						RSB

: Routine Size: 13 bytes, Routine Base: _BASSCODE + 05FE

: 3330
: 3331
: 3333

BASSREC_PROC
1-095

: 2060 3334 1

16-Sep-1984 01:01:12
14-Sep-1984 11:56:35

VAX-11 Bliss-32 V4.0-742
[BASRTL.SRC]BASREC(PO.B32;1)

Page 56
(23)

```

2062      3335 1 GLOBAL ROUTINE BASS$REC_GSE (
2063          3336 1   FOREIGN BUFFER,
2064          3337 1   LOCK FLAGS
2065          3338 1 ) : JSB_D0_READ NOVALUE =
2066          3339 1
2067          3340 1
2068          3341 1
2069          3342 1
2070          3343 1
2071          3344 1
2072          3345 1
2073          3346 1
2074          3347 1
2075          3348 1
2076          3349 1
2077          3350 1
2078          3351 1
2079          3352 1
2080          3353 1
2081          3354 1
2082          3355 1
2083          3356 1
2084          3357 1
2085          3358 1
2086          3359 1
2087          3360 1
2088          3361 1
2089          3362 1
2090          3363 1
2091          3364 1
2092          3365 1
2093          3366 1
2094          3367 1
2095          3368 1
2096          3369 1
2097          3370 1
2098          3371 1
2099          3372 1
2100          3373 1
2101          3374 1
2102          3375 1
2103          3376 1
2104          3377 1
2105          3378 2
2106          3379 2
2107          3380 2
2108          3381 2
2109          3382 2
2110          3383 2
2111          3384 2
2112          3385 2
2113          3386 2
2114          3387 2
2115          3388 2
2116          3389 2
2117          3390 2
2118          3391 2
2119      !+ 1

```

GLOBAL ROUTINE BASS\$REC_GSE (FOREIGN BUFFER,
 LOCK FLAGS) : JSB_D0_READ NOVALUE =

FUNCTIONAL DESCRIPTION:
 Read one record. Update RECOUNT if successful.
 If a foreign buffer is specified, then change RABSL_RBF to point to the
 "foreign buffer". Otherwise, signal a fatal error.

FORMAL PARAMETERS:
 FOREIGN_BUFFER.rl.v points to CB of foreign buffer or 0
 LOCK_FLAGS.rl.u.v bits to set in RAB ROP for manual
 record locking (0 if none)

IMPLICIT INPUTS:
 RABSW_USZ User buffer size of foreign buffer
 RABSL_UBF Pointer to user buffer for foreign buffer
 LUBL_SL_WAIT_TIME Wait time for next input
 WAIT The module level OWN_WAIT

IMPLICIT OUTPUTS:
 RABSB_RAC Record access field
 RECOUNT Own storage for RECOUNT function.
 RABSL_RBF Record pointer in RAB.
 RABSW_RSZ Number of bytes read (stored in RECOUNT)

ROUTINE VALUE:
 NONE

SIDE EFFECTS:
 RABSW_USZ and RABSW_UBF are altered while this routine is running,
 but are restored before exit.
 Reads next record from file on this logical unit.
 SIGNALS any RMS errors

BEGIN

EXTERNAL REGISTER
 CCB : REF BLOCK [, BYTE];

MAP
 FOREIGN_BUFFER : REF BLOCK [, BYTE];

LOCAL
 RMS_STATUS,
 SAVE_USZ,
 ACTUAL_RSZ,
 WAIT_TIME;

! Save the USZ
 Actual record size
 Current wait time

2119 3392 2 | Save USZ in case it is modified. It is faster to always
2120 3393 2 | save and restore it, since that eliminates the test for foreign
2121 3394 2 | buffers and single-character mode at the end of this routine.
2122 3395 2 |
2123 3396 2 | SAVE_USZ = .CCB [RABSW_USZ];
2124 3397 2 |
2125 3398 2 | If a timeout has been specified, store information in the RAB to tell
2126 3399 2 | RMS about it. If no timeout has been specified, clear the TMO bit
2127 3400 2 | in case there was an earlier timeout specified.
2128 3401 2 |
2129 3402 2 |
2130 3403 2 |
2131 3404 2 |
2132 3405 2 | If WAIT is zero then use the LUB's wait. This is to provide upward compatibility
2133 3406 2 | i.e. existing EXE's can run with the LUB wait value in V2.2.
2134 3407 2 |
2135 3408 2 | WAIT_TIME = (IF (.WAIT EQ 0) THEN .CCB [LUBSL_WAIT_TIME] ELSE .WAIT);
2136 3409 2 |
2137 3410 2 | IF (.WAIT_TIME EQ 0)
2138 3411 2 | THEN
2139 3412 2 | CCB [RABSV_TMO] = 0
2140 3413 2 | ELSE
2141 3414 2 | BEGIN
2142 3415 2 | CCB [RABSB_TMO] = .WAIT_TIME;
2143 3416 2 | CCB [RABSV_TMO] = 1;
2144 3417 2 | END;
2145 3418 2 |
2146 3419 2 |
2147 3420 2 | Set the Read-no-echo RMS bit based on the user's last call to
2148 3421 2 | ECHO or NOECHO.
2149 3422 2 |
2150 3423 2 | CCB [RABSV_RNE] = .CCB [LUBSV_NOECHO];
2151 3424 2 |
2152 3425 2 | Set the record pointer field in the RAB to the user buffer. This is
2153 3426 2 | done on each element transmission and not just at OPEN because of RMS
2154 3427 2 | Locate mode.
2155 3428 2 | Fill the input buffer with Nulls. Basic semantics require null fill if
2156 3429 2 | the record read is shorter than the buffer.
2157 3430 2 | Set the record access field in the RAB to sequential. Perform the GET.
2158 3431 2 | If RMS returns a failure status, signal the error. If the GET is
2159 3432 2 | successful, then update recount.
2160 3433 2 |
2161 3434 2 |
2162 3435 2 | IF (.FOREIGN_BUFFER NEQA 0)
2163 3436 2 | THEN
2164 3437 2 | BEGIN
2165 3438 2 |
2166 3439 2 | A foreign buffer was specified. Save off RAB\$L_usz of the "file" channel
2167 3440 2 | and then substitute the appropriate values from the foreign channel into
2168 3441 2 | the file channel to make the record go directly into the foreign buffer.
2169 3442 2 |
2170 3443 2 | CCB [RABSW_USZ] = .FOREIGN_BUFFER [RABSW_USZ];
2171 3444 2 |
2172 3445 2 |
2173 3446 2 |
2174 3447 2 | If the user has called ONECHR, shrink the effective size of the
2175 3448 2 | buffer to one character, so he will get characters one at a time.

```
2176      3449 2 | The user must call ONECHR before each GET, so we clear the ONECHR
2177      3450 2 | flag here.
2178      3451 2 |
2179      3452 2 |
2180      3453 2 |
2181      3454 2 |
2182      3455 2 |
2183      3456 2 |
2184      3457 2 |
2185      3458 2 |
2186      3459 2 |
2187      3460 2 |
2188      3461 2 |
2189      3462 2 |
2190      3463 2 |
2191      3464 2 |
2192      3465 2 |
2193      3466 2 |
2194      3467 2 |
2195      3468 2 |
2196      3469 2 |
2197      3470 2 |
2198      3471 2 |
2199      3472 2 |
2200      3473 2 |
2201      3474 2 |
2202      3475 2 |
2203      3476 2 |
2204      3477 2 |
2205      3478 2 |
2206      3479 2 |
2207      3480 2 |
2208      3481 2 |
2209      3482 2 |
2210      3483 2 |
2211      3484 2 |
2212      3485 2 |
2213      3486 2 |
2214      3487 2 |
2215      3488 2 |
2216      3489 2 |
2217      3490 2 |
2218      3491 2 |
2219      3492 2 |
2220      3493 2 |
2221      3494 2 |
2222      3495 2 |
2223      3496 2 |
2224      3497 2 |
2225      3498 2 |
2226      3499 2 |
2227      3500 2 |
2228      3501 2 |
2229      3502 2 |
2230      3503 2 |
2231      3504 2 |
2232      3505 2 | The user must call ONECHR before each GET, so we clear the ONECHR
| flag here.
|-
| IF (.CCB [LUBSV_ONECHR])
| THEN
| BEGIN
| CCB [LUBSV_ONECHR] = 0;
| CCB [RABSW_USZ] = 1;
| END;
|
| CCB [RAB$B_RAC] = RAB$C_SEQ;
|
|+
| Set bits in the RAB ROP (careful not to turn off ULK).
|
| CCB [RABSL_ROP] = .CCB [RABSL_ROP] OR .LOCK_FLAGS;
|
| RMS_STATUS = $GET (RAB = .CCB);
|
| IF .RMS_STATUS EQL RMSS_CONTROLC
| THEN
|   BAS$SIGNAL_CTRLC ();
|
| IF NOT .RMS_STATUS
| THEN
|   BEGIN
|
|+ We cannot call GET_ERROR because we must restore UBF and USZ.
|-
|
| WHILE (.CCB [RABSL_STS] EQL RMSS_RSA) DO
| BEGIN
| SWAIT (RAB = .CCB);
| $GET (RAB = .CCB);
| END;
|
| END;
|
|+
| Clear RAB ROP bits so that a subsequent I/O operation does not
| inherit them.
|
| CCB [RABSL_ROP] = .CCB [RABSL_ROP] XOR .LOCK_FLAGS;
|
|+
| This actual record size may or may not change below. If the file is a
| terminal device then it will get terminators tacked on to the record read.
|
| ACTUAL_RSZ = .CCB [RABSW_RSZ];
|
|+
| Tack on the terminators when a terminal device file, just like INPUT LINE
|-
```

```
2233      3506    IF .CCB[LUBSV_TERM_DEV]
2234      3507    THEN
2235      3508    BEGIN
2236      3509    LITERAL_K_ESCAPE = XX'1B'.
2237      3510    K_CR = XX'0D'.
2238      3511    K_CRLF = XX'0A0D';
2239
2240
2241      3513    !+ STVO is the escape character, STV2 is the number of terminating characters.
2242      3514    SELECTONEU .CCB[RABSW_STVO] OF
2243      3515    SET
2244      3516    [K_ESCAPE] :
2245      3517    BEGIN
2246      3518
2247      3519    !+ If the length is one then escape is not at the end of the buffer and it
2248      3520    must be moved there, otherwise the escape sequence is at the end of the
2249      3521    buffer following the data.
2250      3522
2251      3523
2252      3524    IF .CCB[RABSW_STV2] EQLU 1
2253      3525    THEN
2254      3526    BEGIN
2255      3527    IF .CCB[RABSW_RSZ] GEQU .CCB[RABSW_USZ]
2256      3528    THEN BAS$$STOP-IO(BASSK_RECFLTOO);
2257      3529    CHSMOVE(1,UPLIT(K_ESCAPE),.CCB[RABSL_UBF] + .CCB[RABSW_RSZ]);
2258      3530    ACTUAL_RSZ = .ACTUAL_RSZ + 1;
2259      3531    END
2260      3532
2261      3533    ELSE
2262      3534    ACTUAL_RSZ = .ACTUAL_RSZ + .CCB[RABSW_STV2];
2263      3535    END;
2264      3536    [K_CR] :
2265      3537    BEGIN
2266      3538    IF .CCB[RABSW_RSZ] + 2 GTRU .CCB[RABSW_USZ]
2267      3539    THEN BAS$$STOP-IO(BASSK_RECFLTOO);
2268      3540    CHSMOVE(2,UPLIT(K_CRLF),.CCB[RABSL_UBF] + .CCB[RABSW_RSZ]);
2269      3541    ACTUAL_RSZ = .ACTUAL_RSZ + 2 ;
2270      3542    END;
2271      3543    [OTHERWISE] :
2272      3544    :
2273      3545    TES;
2274      3546    END;
2275
2276      3548    !+ If there are no errors, null pad the buffer.
2277      3549
2278      3550
2279      3551    IF (.CCB[RABSW_USZ] GTR .ACTUAL_RSZ)AND .CCB[RABSL_STS]
2280      3552    THEN
2281      3553    CHSFILL(XX'00',
2282      3554    .CCB[RABSW_USZ] - .ACTUAL_RSZ, .CCB[RABSL_UBF] + .ACTUAL_RSZ);
2283
2284
2285      3557    !+ Before checking for errors, restore UBF and USZ, and set RECOUNT.
2286      3558
2287      3559    .CCB[RABSL_UBF] = .CCB[LUBSA_UBF];
2288      3560    .CCB[RABSW_USZ] = .SAVE_USZ;
2289      3561    RECOUNT = .ACTUAL_RSZ;
```

```

: 2290      3563 2 1+
: 2291      3564 2 1+ Any error remaining (which will be an error other than Record Stream
: 2292      3565 2 1+ Active, RSA) is fatal.
: 2293      3566 2 1-
: 2294      3567 2 1-
: 2295      3568 2 1 IF ( NOT .CCB [RABSL_STS] ) THEN BASS$STOP_IO (BASSK_IOERR_REC);
: 2296      3569 2 1 CCB [LUBSA_RBUF_ADR] = .CCB [RABSL_RBF];
: 2297      3570 2 1 RETURN;
: 2298      3571 2 1
: 2299      3572 1 1 END;
                                         ! End of BASS$REC_GSE

```

0000001B	0060B	.BLKB	1
00000A0D	0060C P.AAA:	.LONG	27
	00610 P.AAB:	.LONG	2573

.EXTRN SYSSWAIT

3C BB 00000 BASS\$REC_GSE::					
			PUSHR	#^M<R2,R3,R4,R5>	3335
			SUBL2	#16, SP	
			MOVL	R1, R4	
			MOVAB	32(CCB), 8(SP)	
			MOVZWL	28(SP), SAVE_USZ	
			MOVZWL	WAIT, R1	
			BNEQ	18	
			MOVL	-52(CCB), WAIT_TIME	
			MOVAB	4(CCB), R2	
			TSTL	WAIT_TIME	
			BNEQ	28	
			BICB2	#2, 3(R2)	
			BRB	38	
			MOVB	WAIT TIME, 31(CCB)	
			BISB2	#2, 3(R2)	
			INSV	-96(CCB), #0, #1, 3(R2)	
			TSTL	FOREIGN_BUFFER	
			BEQL	48	
			MOVW	32(FOREIGN BUFFER), 28(SP)	
			BBC	#1, -96(CCB), 58	
			BICB2	#2, -96(CCB)	
			MOVW	#1, 28(SP)	
			CLRB	30(CCB)	
			BISL2	LOCK_FLAGS, (R2)	
			PUSHL	CCB	
			CALLS	#1, SYSSGET	
			MOVL	R0, RMS STATUS	
			CMPL	RMS_STATUS, #67153	
			BNEQ	68	
			CALLS	#0, BASS\$SIGNAL_CTRLC	
			BLBS	RMS_STATUS, 88	
			CMPL	8(CCB), #99034	
			BNEQ	88	
			PUSHL	CCB	
			CALLS	#1, SYSSWAIT	
			PUSHL	CCB	
			CALLS	#1, SYSSGET	
03 A2	01	00	A0		
			10	C2 00002	
			51	D0 00005	
	08	AE	20	AB 9E 00008	
	04	AE	08	BE 3C 0000D	
		51	00000000	EF 3C 00012	
				04 12 00019	
		51		AB D0 0001B	
		52		AB 9F 0001F	18:
				51 D5 00023	
				06 12 00025	
		03	A2	02 8A 00027	
				08 11 0002B	
		1F	AB	51 90 0002D	28:
		03	A2	02 88 00031	38:
				50 D5 0003C	
				05 13 0003E	
		08	BE	A0 B0 00040	
		08	AB	01 E1 00045	48:
		A0	AB	02 8A 0004A	
		A0	AB	01 B0 0004E	
		08	BE	1E AB 94 00052	58:
				54 C8 00055	
				58 DD 00058	
		00000000G	00	01 FB 0005A	
				50 D0 00061	
		53		53 D1 00064	
		00010651	8F	07 12 00068	
				00 FB 0006D	
		00000000G	00	53 E8 00074	68:
				08 AB D1 00077	78:
		1E		14 12 0007F	
		000182DA	8F	58 DD 00081	
				01 FB 00083	
		00000000G	00	58 DD 0008A	
				01 FB 0008C	

C 9
16-Sep-1984 01:01:12 VAX-11 Bliss-32 v4.0-742
14-Sep-1984 11:56:35 [BASRTL.SRC]BASRECPRO.B32;1

Page 62
(24)

; Routine Size: 338 bytes, Routine Base: _BASSCODE + 0614

: 2300 3573 1

```

: 2302      3574 1 GLOBAL ROUTINE BASS$REC_GIN (
: 2303          3575 1           ! GET (indexed) a record
: 2304          3576 1           KEY_NO, REL_OP, KEY, FOREIGN_BUFFER, LOCK_FLAGS) : JSB_REC_IND1 NOVALUE =
: 2305          3577 1
: 2306          3578 1 ** FUNCTIONAL DESCRIPTION:
: 2307          3579 1
: 2308          3580 1     Read one record. Update RECOUNT if successful.
: 2309          3581 1     If a foreign buffer is specified, then change RABSL_RBF to point to the
: 2310          3582 1     "foreign buffer". Otherwise, signal a fatal error.
: 2311          3583 1
: 2312          3584 1 ** FORMAL PARAMETERS:
: 2313          3585 1
: 2314          3586 1     KEY_NO.rl.v           key of reference number
: 2315          3587 1     REL_OP.rl.v           relative relationship of the key
: 2316          3588 1     KEY.rt.dx             key to search for
: 2317          3589 1     FOREIGN_BUFFER.rl.v   points to CB of foreign buffer or 0
: 2318          3590 1     LOCK_FLAGS.rlu.v    bits to set in RAB ROP to control manual record
: 2319          3591 1           locking (0 if none)
: 2320          3592 1
: 2321          3593 1 ** IMPLICIT INPUTS:
: 2322          3594 1
: 2323          3595 1     RABSW_USZ            User buffer size of foreign buffer
: 2324          3596 1     RABSL_UBF             Pointer to user buffer for foreign buffer
: 2325          3597 1
: 2326          3598 1 ** IMPLICIT OUTPUTS:
: 2327          3599 1
: 2328          3600 1     RABSB_RAC             Record access field
: 2329          3601 1     RECOUNT               Own storage for RECOUNT function.
: 2330          3602 1     RABSL_RBF             Record pointer in RAB.
: 2331          3603 1     RABSW_USZ             User buffer size for "file" buffer
: 2332          3604 1     RABSL_UBF             Pointer to user buffer for "file" buffer
: 2333          3605 1
: 2334          3606 1 ** ROUTINE VALUE:
: 2335          3607 1     NONE
: 2336          3608 1
: 2337          3609 1
: 2338          3610 1 ** SIDE EFFECTS:
: 2339          3611 1
: 2340          3612 1     Reads next record from file on this logical unit.
: 2341          3613 1
: 2342          3614 1     SIGNALS any RMS errors
: 2343          3615 1
: 2344          3616 2 ** BEGIN
: 2345          3617 2
: 2346          3618 2 ** EXTERNAL REGISTER
: 2347          3619 2     CCB : REF_BLOCK [, BYTE];
: 2348          3620 2
: 2349          3621 2 ** MAP
: 2350          3622 2     KEY : REF_BLOCK [8, BYTE],           ! descriptor of the key
: 2351          3623 2     FOREIGN_BUFFER : REF_BLOCK [, BYTE];
: 2352          3624 2
: 2353          3625 2 ** LITERAL
: 2354          3626 2     K_EQUAL = 0,                      ! keys should be equal
: 2355          3627 2     K_GREATER_EQUAL = 1,           ! key should be greater than or equal
: 2356          3628 2     K_GREATER_THAN = 2;           ! key should be greater than
: 2357          3629 2
: 2358          3630 2 ** LOCAL

```

: 2359 3631 2 RMS STATUS.
: 2360 3632 2
: 2361 3633 2
: 2362 3634 2
: 2363 3635 2
: 2364 3636 2
: 2365 3637 2
: 2366 3638 2
: 2367 3639 2
: 2368 3640 2
: 2369 3641 2
: 2370 3642 2
: 2371 3643 2
: 2372 3644 2
: 2373 3645 2
: 2374 3646 2
: 2375 3647 2
: 2376 3648 2
: 2377 3649 2
: 2378 3650 2
: 2379 3651 2
: 2380 3652 2
: 2381 3653 2
: 2382 3654 2
: 2383 3655 2
: 2384 3656 2
: 2385 3657 2
: 2386 3658 2
: 2387 3659 2
: 2388 3660 2
: 2389 3661 2
: 2390 3662 2
: 2391 3663 2
: 2392 3664 2
: 2393 3665 2
: 2394 3666 2
: 2395 3667 2
: 2396 3668 2
: 2397 3669 2
: 2398 3670 2
: 2399 3671 2
: 2400 3672 2
: 2401 3673 2
: 2402 3674 2
: 2403 3675 2
: 2404 3676 2
: 2405 3677 2
: 2406 3678 2
: 2407 3679 2
: 2408 3680 2
: 2409 3681 2
: 2410 3682 2
: 2411 3683 2
: 2412 3684 2
: 2413 3685 2
: 2414 3686 2
: 2415 3687 2

RMS STATUS.
SAVE_USZ;
! Save the USZ

Save USZ in case it is modified. It is faster to always save and restore it, since that eliminates the test for foreign buffers and single-character mode at the end of this routine.

SAVE_USZ = .CCB [RABSW_USZ];

Set the record pointer field in the RAB to the user buffer. This is done on each element transmission and not just at OPEN because of RMS Locate mode.

Fill the input buffer with Nulls. Basic semantics require null fill if the record read is shorter than the buffer.

Set the record access field in the RAB to sequential. Perform the GET. If RMS returns a failure status, signal the error. If the GET is successful, then update recount.

IF .FOREIGN_BUFFER NEQ 0
THEN
BEGIN

A foreign buffer was specified. Substitute the appropriate values from the foreign channel into the file channel to make the record go directly into the foreign buffer.

.CCB [RABSL_UBF] = .FOREIGN_BUFFER [RABSL_UBF];
.CCB [RABSW_USZ] = .FOREIGN_BUFFER [RABSW_USZ];
END;

Set the record access field to key. Set KBF to the key. Set KSZ to the size of the key passed. Set the key of reference to the desired key. Use a case statement to toggle KGT and KGE in the ROP.

.CCB [RABSB_RAC] = RABSC_KEY;
.CCB [RABSL_KBF] = .KEY [DSC\$A_POINTER];
.CCB [RABSB_KRF] = .KEY_NO;
.CCB [RABSB_KSZ] = {IF .KEY [DSC\$B_DTYPE] NEQ DSC\$K_DTYPE_P
THEN
.KEY [DSC\$W_LENGTH]
ELSE
.KEY [DSC\$W_LENGTH]/2 + 1});

CASE .REL_OP FROM K_EQUAL TO K_GREATER_THAN OF
SET

[K_EQUAL] :
.CCB [RABSV_KGE] = CCB [RABSV_KGT] = 0;

[K_GREATER_EQUAL] :
BEGIN
.CCB [RABSV_KGE] = 1;
.CCB [RABSV_KGT] = 0;

```
2416      3688 2      END;  
2417      3689  
2418      3690  
2419      3691  
2420      3692 [K_GREATER_THAN] :  
2421      3693     BEGIN  
2422      3694     [CCB [RAB$V_KGT] = 1;  
2423      3695     [CCB [RAB$V_KGE] = 0;  
2424      3696     END;  
2425      3697     TES;  
2426      3698 1+ Set bits in RAB ROP without turning off ULK.  
2427      3699 1-  
2428      3700     CCB [RAB$L_ROP] = .CCB [RAB$L_ROP] OR .LOCK_FLAGS;  
2429      3701     RMS_STATUS = SGET (RAB = .CCB);  
2430      3702     IF .RMS_STATUS EQ RMSS_CONTROLC  
2431      3703     THEN  
2432      3704     BAS$SIGNAL_CTRLC ();  
2433      3705  
2434      3706  
2435      3707  
2436      3708  
2437      3709  
2438      3710  
2439      3711 1+ We cannot call GET_ERROR because we must restore UBF and USZ.  
2440      3712 1-  
2441      3713     WHILE (.CCB [RAB$L_STS] EQ RMSS_RSA) DO  
2442      3714     BEGIN  
2443      3715     SWAIT (RAB = .CCB);  
2444      3716     SGET (RAB = .CCB);  
2445      3717  
2446      3718  
2447      3719  
2448      3720     END;  
2449      3721  
2450      3722  
2451      3723  
2452      3724  
2453      3725 1+ Turn off bits in RAB ROP so that subsequent I/O operations can not  
2454      3726 inherit them.  
2455      3727 1-  
2456      3728     CCB [RAB$L_ROP] = .CCB [RAB$L_ROP] XOR .LOCK_FLAGS;  
2457      3729  
2458      3730  
2459      3731  
2460      3732 1+ If there are no errors, null pad the buffer.  
2461      3733 1-  
2462      3734  
2463      3735     IF (.CCB [RAB$W_USZ] GTR .CCB [RAB$W_RSZ]) AND .CCB [RAB$L_STS]  
2464      3736     THEN  
2465      3737     CHSFILL (IX'00,  
2466      3738     .CCB [RAB$W_USZ] - .CCB [RAB$W_RSZ], .CCB [RAB$L_UBF] + .CCB [RAB$W_RSZ]);  
2467      3739  
2468      3740  
2469      3741 1+ Before checking for errors, restore UBF and USZ, and set RECOUNT.  
2470      3742 1-  
2471      3743     CCB [RAB$L_UBF] = .CCB [LUB$A_UBF];  
2472      3744     CCB [RAB$W_USZ] = .SAVE_USZ;
```

```

: 2473      3745 2 RECOUNT = .CCB [RABSW_RSZ];
: 2474      3746
: 2475      3747
: 2476      3748 Any error remaining (which will be an error other than Record Stream
: 2477      3749 Active, RSA) is fatal.
: 2478      3750
: 2479      3751 IF ( NOT .CCB [RABSL_STS] ) THEN BASS$STOP_IO (BASSK_IOERR_REC);
: 2480      3752
: 2481      3753
: 2482      3754 This is frosting on the cake. LUBSA_RBUF_ADR points to the record buffer for
: 2483      3755 MOVE. The buffer may change due to RMS Locate Mode. Currently, Locate Mode
: 2484      3756 is not performed on files which UPDATE or PUT. However, in anticipation that
: 2485      3757 RMS may add such a capability, we point RBUF_ADR off to the buffer used by PUT.
: 2486      3758
: 2487      3759 [CB [LUBSA_RBUF_ADR] = .CCB [RABSL_RBF];
: 2488      3760 RETURN;
: 2489      3761 END;

```

! End of BASS\$REC_GIN

3C BB 00000 BASS\$REC_GIN::										
	7E	20	AB	3C 00002	PUSHR	#M<R2,R3,R4,R5>				3574
			53	D5 00006	MOVZWL	32(CCB), SAVE USZ				3639
			0A	13 00008	TSTL	FOREIGN_BUFFER				3652
24	AB	24	A3	D0 0000A	BEQL	18				3659
20	AB	20	A3	B0 0000F	MOVL	36(FOREIGN_BUFFER), 36(CCB)				3660
1E	AB	01	90	00014	MOVW	32(FOREIGN_BUFFER), 32(CCB)				3669
30	AB	04	A2	D0 00018	1\$: MOVB	#1, 30(CCB)				3670
35	AB	50	90	0001D	MOVL	4(KEY), 48(CCB)				3671
		15	02	A2 91 00021	MOV8	KEY NO, 53(CCB)				3672
			05	13 00025	CMPB	2(KEY), #21				
			52	62 3C 00027	BEQL	28				
				08 11 0002A	MOVZWL	(KEY), R2				3674
			52	62 3C 0002C	BRB	38				
			52	02 C6 0002F	28\$: DIVL2	(KEY), R2				3676
			52	D6 00032	INCL	#2, R2				
		34	AB	52 90 00034	38\$: MOVB	R2, 52(CCB)				3672
			52	AB 9E 00038	MOVAB	4(CCB), R2				3682
02	00	04	51	CF 0003C	CASEL	REL OP, #0, #2				3678
0018		0000D		00006	.WORD	58-48,-				
				00040	48\$: BICB2	68-48,-				
			02	A2 40	58: BRB	78-48				3682
			02	A2 10	11 0004B	88				
			02	A2 20	88 0004D	68\$: BISB2	#32, 2(R2)			3686
			02	A2 09	8A 00051	BICB2	#64, 2(R2)			3687
			02	A2 09	11 00056	BRB	98			3678
			02	A2 40	88 00058	78\$: BISB2	#64, 2(R2)			3692
			02	A2 20	8A 0005D	88\$: BICB2	#32, 2(R2)			3693
			62	54 C8 00061	98\$: BISL2	LOCK_FLAGS, (R2)				3701
				58 DD 00064	PUSHL	CCB				3703
00000000G	00			01 FB 00066	CALLS	#1, SYSSGET				
			53	50 D0 0006D	MOVL	RO, RMS STATUS				
00010651	8F			53 D1 00070	CMPL	RMS_STATUS, #67153				3705

			07	12	00077	BNEQ	10\$			3707	
			00	FB	00079	CALLS	#0	BASS\$SIGNAL_CTRLC		3709	
			53	E8	00080	10\$:	BLBS	RM\$ STATUS 12\$		3716	
		08	AB	D1	00083	11\$:	CMPL	8(CC8), #99034		3718	
			14	12	00088	BNEQ	12\$			3719	
			58	DD	0008D	PUSHL	CC8			3729	
			01	FB	0008F	CALLS	#1	SYSSWAIT		3735	
			58	DD	00096	PUSHL	CC8			3738	
			01	FB	00098	CALLS	#1	SYSSGET		3745	
			E2	11	0009F	BRB	11\$			3751	
			54	CC	000A1	12\$:	XORL2	LOCK FLAGS, (R2)		3759	
	22	AB	20	AB	B1	000A4	CMPW	32(CC8), 34(CC8)		3761	
			1D	1B	000A9	BLEQU	13\$				
			08	AB	F9	000AB	BLBC	8(CC8), 13\$			
			20	AB	3C	000AF	MOVZWL	32(CC8), R1			
			22	AB	3C	000B3	MOVZWL	34(CC8), R0			
			50	C2	000B7	SUBL2	R0, R1				
			50	AB	3C	000BA	MOVZWL	34(CC8), R0			
			50	AB	C0	000BE	ADDL2	36(CC8), R0			
51	00		24	AB	00	2C 000C2	MOVC5	#0, (SP), #0, R1, (R0)			
			6E		60	000C7					
			9C	AB	D0	000C8	13\$:	MOVL	-100(CC8), 36(CC8)		
			20	AB	6E	000CD	MOVW	SAVE USZ, 32(CC8)		3743	
			EF	AB	22	AB 000D1	MOVZWL	34(CC8), RECOUNT		3744	
			0A	08	AB	E8 000D9	BLBS	8(CC8), 14\$		3745	
			7E	01	CE	000DD	MNEGL	#1, -(SP)		3751	
			00	01	FB	000E0	CALLS	#1, BASS\$STOP IO			
			EC	AB	28	AB 000E7	14\$:	MOVL	40(CC8), -20(CC8)		3759
			5E		04	C0 000EC	ADDL2	#4, SP		3761	
					3C	BA 000EF	POPR	#^M<R2,R3,R4,R5>			
					05	000F1	RSB				

: Routine Size: 242 bytes, Routine Base: _BASSCODE + 0766

: 2490 3762 1

2492 3763 1 GLOBAL ROUTINE BASSREC_GRE {
2493 3764 1 FOREIGN_BUFFER,LOCK_FLAGS) : JSB_DO_READ NOVALUE = ! GET (relative) a record
2494 3765 1
2495 3766 1
2496 3767 1
2497 3768 1
2498 3769 1
2499 3770 1
2500 3771 1
2501 3772 1
2502 3773 1
2503 3774 1
2504 3775 1
2505 3776 1
2506 3777 1
2507 3778 1
2508 3779 1
2509 3780 1
2510 3781 1
2511 3782 1
2512 3783 1
2513 3784 1
2514 3785 1
2515 3786 1
2516 3787 1
2517 3788 1
2518 3789 1
2519 3790 1
2520 3791 1
2521 3792 1
2522 3793 1
2523 3794 1
2524 3795 1
2525 3796 1
2526 3797 1
2527 3798 1
2528 3799 1
2529 3800 2
2530 3801 2
2531 3802 2
2532 3803 2
2533 3804 2
2534 3805 2
2535 3806 2
2536 3807 2
2537 3808 2
2538 3809 2
2539 3810 2
2540 3811 2
2541 3812 2
2542 3813 2
2543 3814 2
2544 3815 2
2545 3816 2
2546 3817 2
2547 3818 2
2548 3819 2

GLOBAL ROUTINE BASSREC_GRE {
 FOREIGN_BUFFER,LOCK_FLAGS) : JSB_DO_READ NOVALUE = ! GET (relative) a record
++
FUNCTIONAL DESCRIPTION:
 Read one record from a relative file. Modify the RAB if necessary for foreign buffers. Update RECOUNT if successful. Otherwise, signal a fatal error.
FORMAL PARAMETERS:
 FOREIGN_BUFFER.rl.v The address of the CB of a foreign buffer or 0
 LOCK_FLAGS.rlu.v bits to set in the RAB ROP to control manual record locking (0 if none)
IMPLICIT INPUTS:
 RABSW_RSZ record size for foreign buffer
 RABSL_UBF buffer address for foreign buffer
IMPLICIT OUTPUTS:
 RAB\$B_RAC Record access field
 RECOUNT Own storage for RECOUNT function.
 RABSL_RBF Record pointer in RAB.
 RABSW_USZ record size of file buffer
 RABSL_UBF address of buffer for file buffer
ROUTINE VALUE:
 NONE
SIDE EFFECTS:
 SIGNALS any RMS errors
--
BEGIN
 EXTERNAL REGISTER
 CCB : REF BLOCK [. BYTE];
 MAP
 FOREIGN_BUFFER : REF BLOCK [. BYTE];
 LOCAL
 RMS_STATUS.
 SAVE_USZ: ! Save the USZ
 + Save USZ in case it is modified. It is faster to always save and restore it, since that eliminates the test for foreign buffers and single-character mode at the end of this routine.
 - SAVE_USZ = .CCB [RABSW_USZ];
 !+

1+ Set the record pointer field in the RAB to the user buffer. This is
done on each element transmission and not just at OPEN because of RMS
Locate mode.
1+ Fill the input buffer with Nulls. Basic semantics require null fill if
the record read is shorter than the buffer.
1+ Set the record access field in the RAB to sequential. Perform the GET.
1+ If RMS returns a failure status, signal the error. If the GET is
successful, then update recount.
1-
1+ IF .FOREIGN_BUFFER NEQ 0
1+ THEN BEGIN
1+ There is a foreign buffer. Modify the file buffer to point to the
1+ buffer associated with the foreign buffer's channel.
1+ CCB [RABSL_RBF] = CCB [RABSL_UBF] = .FOREIGN_BUFFER [RABSL_UBF];
1+ CCB [RABSW_RSZ] = CCB [RABSW_USZ] = .FOREIGN_BUFFER [RABSW_USZ];
1+ END
1+ ELSE CCB [RABSL_RBF] = .CCB [RABSL_UBF];
1+ CCB [RABSB_RAC] = RABSC_KEY;
1+ Set bits in RAB ROP without destroying ULK.
1-
1+ CCB [RABSL_ROP] = .CCB [RABSL_ROP] OR .LOCK_FLAGS;
1+ RMS_STATUS = \$GET (RAB = .CCB);
1+ IF .RMS_STATUS EQL RMSS_CONTROLC
1+ THEN BAS\$SIGNAL_CTRLC ();
1+ IF NOT .RMS_STATUS
1+ THEN BEGIN
1+ We cannot call GET_ERROR because we must restore UBF and USZ.
1-
1+ WHILE (.CCB [RABSL_STS] EQL RMSS_RSA) DO
1+ BEGIN
1+ SWAIT (RAB = .CCB);
1+ \$GET (RAB = .CCB);
1+ END;
1+ END;
1+ Turn off bits in the RAB ROP so that subsequent I/O operations can not
1+ inherit them.
1-
1+ 3820 2
1+ 3821
1+ 3822
1+ 3823
1+ 3824
1+ 3825
1+ 3826
1+ 3827
1+ 3828
1+ 3829
1+ 3830
1+ 3831
1+ 3832
1+ 3833
1+ 3834
1+ 3835
1+ 3836
1+ 3837
1+ 3838
1+ 3839
1+ 3840
1+ 3841
1+ 3842
1+ 3843
1+ 3844
1+ 3845
1+ 3846
1+ 3847
1+ 3848
1+ 3849
1+ 3850
1+ 3851
1+ 3852
1+ 3853
1+ 3854
1+ 3855
1+ 3856
1+ 3857
1+ 3858
1+ 3859
1+ 3860
1+ 3861
1+ 3862
1+ 3863
1+ 3864
1+ 3865
1+ 3866
1+ 3867
1+ 3868
1+ 3869
1+ 3870
1+ 3871
1+ 3872
1+ 3873
1+ 3874
1+ 3875
1+ 3876

```

2606      3877 2   CCB [RABSL_ROP] = .CCB [RABSL_ROP] XOR .LOCK_FLAGS;
2607      3878
2608      3879
2609      3880
2610      3881
2611      3882
2612      3883
2613      3884
2614      3885
2615      3886
2616      3887
2617      3888
2618      3889
2619      3890
2620      3891
2621      3892
2622      3893
2623      3894
2624      3895
2625      3896
2626      3897
2627      3898
2628      3899
2629      3900
2630      3901
2631      3902
2632      3903
2633      3904
2634      3905
2635      3906
2636      3907 1   If there are no errors, null pad the buffer.

      IF (.CCB [RABSW_USZ] GTR .CCB [RABSW_RSZ]) AND .CCB [RABSL_STS]
      THEN
          CHSFILL ('XX'00',
          .CCB [RABSW_USZ] - .CCB [RABSW_RSZ], .CCB [RABSL_UBF] + .CCB [RABSW_RSZ]);

      Before checking for errors, restore UBF and USZ, and set RECOUNT.

      CCB [RABSL_UBF] = .CCB [LUBSA_UBF];
      CCB [RABSW_USZ] = .SAVE_USZ;
      RECOUNT = .CCB [RABSW_RSZ];

      Any error remaining (which will be an error other than Record Stream
      Active, RSA) is fatal.

      IF ( NOT .CCB [RABSL_STS]) THEN BASS$STOP_IO (BASS$IOERR_REC);

      Set LUBSA_RBUF ADR to point to the buffer used by RMS. It may move around
      due to Locate Mode.

      CCB [LUBSA_RBUF_ADR] = .CCB [RABSL_RBF];
      RETURN;
      END;

```

! End of BASS\$REC_GRE

			3C BB 00000 BASS\$REC GRE::			
			PUSHR #M<R2,R3,R4,R5>			: 3763
	52	20	51 D0 00002	MOVL R1, R2		
	7E		AB 3C 00005	MOVZWL 32(CCB), SAVE_USZ		
			50 D5 00009	TSTL FOREIGN_BUFFER		
			1A 13 0000B	BEOI 1\$		
	51	24	A0 D0 0000D	MOVL 36(FOREIGN BUFFER), R1		
	24	AB	51 D0 00011	MOVL R1, 36(CCB)		
	28	AB	51 D0 00015	MOVL R1, 40(CCB)		
	50	20	A0 3C 00019	MOVZWL 32(FOREIGN BUFFER), R0		
	20	AB	50 B0 0001D	MOVW R0, 32(CCB)		
	22	AB	50 B0 00021	MOVW R0, 34(CCB)		
			05 11 00025	BRB 2\$		
	28	AB	AB D0 00027	MOVL 36(CCB), 40(CCB)		
	1E	AB	01 90 0002C	MOVB #1, 30(CCB)		
	04	AB	52 C8 00030	BISL2 LOCK_FLAGS, 4(CCB)		
			5B DD 00034	PUSHL CCB		
	0000000G	00	01 FB 00036	CALLS #1, SYSS\$GET		
			50 D0 0003D	MOVL R0, RMS_STATUS		
	00010651	53	53 D1 00040	CMPL RM\$_STATUS, #67153		
			07 12 00047	BNEQ 38		
						3853

L 9
16-Sep-1984 01:01:12 VAX-11 Bliss-32 V4.0-742
14-Sep-1984 11:56:35 [BASRTL.SRC]BASREC PRO.B32;1

	00000000G	00	00	FB	00049		CALLS	#0, BASS\$SIGNAL_CTRLC	3855
	000182DA	1E	53	E8	00050	35:	BLBS	RMS STATUS, \$S	3857
		8F	AB	D1	00053	48:	CMPL	8(CC(B)), #99034	3864
			14	12	0005B		BNEQ	\$S	
			5B	DD	0005D		PUSHL	CC(B)	3866
	00000000G	00	01	FB	0005F		CALLS	#1, SYSSWAIT	3867
			5B	DD	00066		PUSHL	CC(B)	
	00000000G	00	01	FB	00068		CALLS	#1, SYSSGET	3868
			F2	11	0006F		BRB	48	
	04	AB	52	CC	00071	58:	XORL2	LOCK FLAGS, 4(CC(B))	3869
	22	AB	20	AB	B1	00075	CMPW	32(CC(B)), 34(CC(B))	3877
			1D	1B	0007A		BLEQU	6S	3883
			19	08	AB	E9	BLBC	8(CC(B)), 6S	
			51	20	AB	3C	MOVZWL	32(CC(B)), R1	3886
			50	22	AB	3C	MOVZWL	34(CC(B)), R0	
			51	50	C2	00084	SUBL2	R0, R1	
			50	22	AB	3C	MOVZWL	34(CC(B)), R0	
			50	24	AB	C0	ADDL2	36(CC(B)), R0	
51	00	6E	00	2C	00093		MOVCS	#0, (SP), #0, R1, (R0)	
			60	00	00095				
	24	AB	9C	AB	D0	00099	MOVL	-100(CC(B)), 36(CC(B))	3891
	20	AB	6E	AB	B0	0009E	MOVW	SAVE USZ, 32(CC(B))	3892
	00000000	EF	22	AB	3C	000A2	MOVZWL	34(CC(B)), RECOUNT	3893
		0A	08	AB	E8	000AA	BLBS	8(CC(B)), 7S	3899
			01	CE	000AE		MNEG	#1, -(SP)	
	00000000G	7E	01	FB	000B1		CALLS	#1, BASS\$STOP IO	3905
		EC	AB	28	AB	D0	MOVL	40(CC(B)), -20(CC(B))	3907
		5E	04	CO	000BD		ADDL2	#4, SP	
			3C	BA	000C0		POPR	#^H<R2,R3,R4,R5>	
			05	00	000C2		RSB		

; Routine Size: 195 bytes, Routine Base: _BASSCODE + 0858

```

2638 3908 1 GLOBAL ROUTINE BASS$REC_GRFA ( ! GET (by RFA) a record
2639 3909 1 FOREIGN_BUFFER, _LOCK_FLAGS) : JSB_DO_READ NOVALUE =
2640 3910 1
2641 3911 1 ++
2642 3912 1 FUNCTIONAL DESCRIPTION:
2643 3913 1
2644 3914 1 Read one record from a file. Modify the RAB if necessary for
2645 3915 1 foreign buffers. Update RECOUNT if successful. Otherwise, signal a fatal error.
2646 3916 1
2647 3917 1 FORMAL PARAMETERS:
2648 3918 1
2649 3919 1 FOREIGN_BUFFER.rl.v The address of the CB of a foreign
2650 3920 1
2651 3921 1 LOCK_FLAGS.rlu.v bits to set in the RAB ROP to control
2652 3922 1 manual record locking (0 if none)
2653 3923 1
2654 3924 1 IMPLICIT INPUTS:
2655 3925 1 RABSW_RSZ record size for foreign buffer
2656 3926 1 RABSL_UBF buffer address for foreign buffer
2657 3927 1
2658 3928 1 IMPLICIT OUTPUTS:
2659 3929 1
2660 3930 1 RAB$B_RAC Record access field
2661 3931 1 RECOUNT Own storage for RECOUNT function.
2662 3932 1 RABSL_RBF Record pointer in RAB.
2663 3933 1 RABSW_USZ record size of file buffer
2664 3934 1 RABSL_UBF address of buffer for file buffer
2665 3935 1
2666 3936 1 ROUTINE VALUE:
2667 3937 1
2668 3938 1
2669 3939 1
2670 3940 1 SIDE EFFECTS:
2671 3941 1
2672 3942 1 -- SIGNALS any RMS errors
2673 3943 1
2674 3944 1
2675 3945 2 BEGIN
2676 3946 2
2677 3947 2 EXTERNAL REGISTER
2678 3948 2 CCB : REF BLOCK [, BYTE];
2679 3949 2
2680 3950 2
2681 3951 2 MAP FOREIGN_BUFFER : REF BLOCK [, BYTE];
2682 3952 2
2683 3953 2 LOCAL
2684 3954 2 RMS_STATUS,
2685 3955 2 SAVE_USZ; ! Save the USZ
2686 3956 2
2687 3957 2
2688 3958 2 ++
2689 3959 2 Save USZ in case it is modified. It is faster to always
2690 3960 2 save and restore it, since that eliminates the test for foreign
2691 3961 2 buffers and single-character mode at the end of this routine.
2692 3962 2 -
2693 3963 2 SAVE_USZ = .CCB [RABSW_USZ];
2694 3964 2 ++

```

```
2695    3965 2 | Set the record pointer field in the RAB to the user buffer. This is
2696    3966 2 | done on each element transmission and not just at OPEN because of RMS
2697    3967 2 | Locate mode.
2698    3968 2 | Fill the input buffer with Nulls. Basic semantics require null fill if
2699    3969 2 | the record read is shorter than the buffer.
2700    3970 2 | Set the record access field in the RAB to sequential. Perform the GET.
2701    3971 2 | If RMS returns a failure status, signal the error. If the GET is
2702    3972 2 | successful, then update recount.
2703    3973 2 |
2704    3974 2 |
2705    3975 2 | IF .FOREIGN_BUFFER NEQ 0
2706    3976 2 | THEN
2707    3977 2 | BEGIN
2708    3978 2 |
2709    3979 2 | There is a foreign buffer. Modify the file buffer to point to the
2710    3980 2 | buffer associated with the foreign buffer's channel.
2711    3981 2 |
2712    3982 2 | CCB [RAB$L_RBF] = CCB [RAB$L_UBF] = .FOREIGN_BUFFER [RAB$L_UBF];
2713    3983 2 | CCB [RAB$W_RSZ] = CCB [RAB$W_USZ] = .FOREIGN_BUFFER [RAB$W_USZ];
2714    3984 2 | END
2715    3985 2 | ELSE
2716    3986 2 | CCB [RAB$L_RBF] = .CCB [RAB$L_UBF];
2717    3987 2 |
2718    3988 2 | CCB [RAB$B_RAC] = RAB$C_RFA;
2719    3989 2 |
2720    3990 2 | Set bits in RAB ROP without destroying ULK.
2721    3991 2 |
2722    3992 2 |
2723    3993 2 |
2724    3994 2 | CCB [RAB$L_ROP] = .CCB [RAB$I_ROP] OR .LOCK_FLAGS;
2725    3995 2 |
2726    3996 2 | RMS_STATUS = $GET (RAB = .CCB);
2727    3997 2 |
2728    3998 2 | IF .RMS_STATUS EQL RMSS_CONTROLC
2729    3999 2 | THEN
2730    4000 2 |   BASS$SIGNAL_CTRLC ();
2731    4001 2 |
2732    4002 2 | IF NOT .RMS_STATUS
2733    4003 2 | THEN
2734    4004 2 | BEGIN
2735    4005 2 |
2736    4006 2 | We cannot call GET_ERROR because we must restore UBF and USZ.
2737    4007 2 |
2738    4008 2 |
2739    4009 2 | WHILE (.CCB [RAB$L_STS] EQL RMSS_RSA) DO
2740    4010 2 | BEGIN
2741    4011 2 |   SWAIT (RAB = .CCB);
2742    4012 2 |   $GET (RAB = .CCB);
2743    4013 2 | END;
2744    4014 2 |
2745    4015 2 |
2746    4016 2 |
2747    4017 2 |
2748    4018 2 | Turn off bits in the RAB ROP so that subsequent I/O operations can not
2749    4019 2 | inherit them.
2750    4020 2 |
2751    4021 2 |
```

```

: 2752    4022 2   CCB [RABSL_ROP] = .CCB [RABSL_ROP] XOR .LOCK_FLAGS;
: 2753    4023
: 2754    4024
: 2755    4025
: 2756    4026
: 2757    4027
: 2758    4028
: 2759    4029
: 2760    4030
: 2761    4031
: 2762    4032
: 2763    4033
: 2764    4034
: 2765    4035
: 2766    4036
: 2767    4037
: 2768    4038
: 2769    4039
: 2770    4040
: 2771    4041
: 2772    4042
: 2773    4043
: 2774    4044
: 2775    4045
: 2776    4046
: 2777    4047
: 2778    4048
: 2779    4049
: 2780    4050
: 2781    4051
: 2782    4052 1

      CCB [RABSW_USZ] GTR .CCB [RABSW_RSZ]) AND .CCB [RABSL_STS]
      THEN
          CHSFILL (ZX'00',
          .CCB [RABSW_USZ] - .CCB [RABSW_RSZ], .CCB [RABSL_UBF] + .CCB [RABSW_RSZ]);

      Before checking for errors, restore UBF and USZ, and set RECOUNT.
      CCB [RABSL_UBF] = .CCB [LUBSA_UBF];
      CCB [RABSW_USZ] = .SAVE_USZ;
      RECOUNT = .CCB [RABSW_RSZ];

      Any error remaining (which will be an error other than Record Stream
      Active, RSA) is fatal.

      IF ( NOT .CCB [RABSL_STS]) THEN BASS$STOP_IO (BASSK_IOERR_REC);

      Set LUBSA_RBUF_ADR to point to the buffer used by RMS. It may move around
      due to Locate Mode.

      CCB [LUBSA_RBUF_ADR] = .CCB [RABSL_RBF];
      RETURN;
      END;

```

! End of BASS\$REC_GRFA

			3C BB 00000 BASS\$REC GRFA::			
	52	20	PUSHR #^M<R2,R3,R4,R5>			3908
	7E		MOVL R1, R2			3962
			MOVZWL 32(CC), SAVE_USZ			3975
			TSTL FOREIGN_BUFFER			
	51	24	BEQL 1\$			
24	AB		MOVL 36(FOREIGN_BUFFER), R1			3982
28	AB		MOVL R1, 36(CC)			
	51	20	MOVL R1, 40(CC)			
	50		MOVZWL 32(FOREIGN_BUFFER), R0			3983
	AB		MOVW R0, 32(CC)			
20	AB		MOVW R0, 34(CC)			
22	AB		BRB 2\$			3975
	50		MOVL 36(CC), 40(CC)			3986
	50		MOVB #2, 30(CC)			3988
	50		BISL2 LOCK_FLAGS, 4(CC)			3994
	50		PUSHL CCB			3996
04	AB		CALLS #1, SYSSGET			
	52		MOVL R0, RMS_STATUS			
	52		CMPBL RMS_STATUS, #67153			
00000000G	00		BNEQ 38			3998
00010651	0F					
	53					
	53					
	07					
	12					
	00047					

00000000G	00		00	FB	00049		CALLS	#0, BASS\$SIGNAL_CTRLC	4000
000182DA	1E		53	E8	00050	38:	BLBS	RM\$ STATUS 58	4002
	BF	08	AB	D1	00053	48:	(MPL	8(CCB), #99034	4009
			14	12	0005B		BNEQ	58	
			5B	DD	0005D		PUSHL	CC8	
00000000G	00		01	FB	0005F		CALLS	#1, SYSSWAIT	4011
00000000G	00		5B	DD	00066		PUSHL	CC8	4012
			01	FB	00068		CALLS	#1, SYSSGET	
			E2	11	0006F		BRB	48	
	04	AB	52	CC	00071	58:	XORL2	LOCK FLAGS, 4(CCB)	4009
	22	AB	20	AB	B1	00075	(MPW	32(CCB), 34(CCB)	4022
			1D	1B	0007A		BLEQU	65	4028
			19	08	AB	F9	BLBC	8(CCB), 68	
			51	20	AB	3C	MOVZWL	32(CCB), R1	4031
			50	22	AB	3C	MOVZWL	34(CCB), R0	
			51	50	50	C2	SUBL2	R0, R1	
			50	22	AB	3C	MOVZWL	34(CCB), R0	
			50	24	AB	C0	ADDL2	36(CCB), R0	
51	00		6E	00	2C	00093	MOVC5	#0, (SP), #0, R1, (R0)	
				60		00098			
			24	AB	9C	AB	MOVL	-100(CCB), 36(CCB)	4036
	20	AB	6E	6E	B0	0009E	MOVW	SAVE USZ, 32(CCB)	4037
00000000	EF		22	AB	3C	000A2	MOVZWL	34(CCB), RECOUNT	4038
	0A	08	AB	E8	000AA		BLBS	8(CCB), 78	4044
			7E	01	CE	000AE	MNEG	#1, -(SP)	
00000000G	00		01	FB	000B1		CALLS	#1, BASS\$STOP IO	
	EC	AB	28	AB	D0	000B8	MOVL	40(CCB), -20(CCB)	4050
			5E	04	C0	000BD	ADDL2	#4, SP	4052
				3C	BA	000C0	POPR	#^M<R2,R3,R4,R5>	
				05	000C2		RSB		

: Routine Size: 195 bytes. Routine Base: _BASS\$CODE + 0918

: 2783 4053 1
: 2784 4054 1

2786 4055 1 GLOBAL ROUTINE BASSREC_PSE (! PUT (sequential) a record
2787 4056 1 COUNT ! No. of bytes to write
2788 4057 1 FOREIGN_BUFFER ! pointer to foreign buffer (B or 0
2789 4058 1) : JSB_PUT NOVALUE =
2790 4059 1
2791 4060 1 **
2792 4061 1 FUNCTIONAL DESCRIPTION:
2793 4062 1
2794 4063 1 Check for "foreign buffers" and point RABSL_RSZ to foreign USZ if there.
2795 4064 1 Write one record. If successful then return; otherwise, signal a fatal
2796 4065 1 error.
2797 4066 1
2798 4067 1 FORMAL PARAMETERS:
2799 4068 1
2800 4069 1 COUNT.rl.v No. of bytes to write
2801 4070 1 FOREIGN_BUFFER.rl.v pointer to foreign buffer (B or 0
2802 4071 1
2803 4072 1 IMPLICIT INPUTS:
2804 4073 1
2805 4074 1 RABSW_RSZ of foreign buffer
2806 4075 1 RABSL_RBF of foreign buffer
2807 4076 1 LUBSV_CCO Cancel control 0
2808 4077 1
2809 4078 1 IMPLICIT OUTPUTS:
2810 4079 1
2811 4080 1 RABSL_RBF for "file" buffer
2812 4081 1 RABSW_RSZ length of record to write
2813 4082 1 LUBSL_LOG_RECNO logical record number
2814 4083 1 RABSB_RAC record access field
2815 4084 1 RABSV_CCO Cancel control 0
2816 4085 1
2817 4086 1 ROUTINE VALUE:
2818 4087 1
2819 4088 1
2820 4089 1
2821 4090 1
2822 4091 1
2823 4092 1
2824 4093 1
2825 4094 1
2826 4095 2
2827 4096 2
2828 4097 2
2829 4098 2
2830 4099 2
2831 4100 2
2832 4101 2
2833 4102 2
2834 4103 2
2835 4104 2
2836 4105 2
2837 4106 2
2838 4107 2
2839 4108 2
2840 4109 2
2841 4110 2
2842 4111 2
SIDE EFFECTS:
-- SIGNALS any RMS errors

BEGIN
EXTERNAL REGISTER
CCB : REF BLOCK [, BYTE];
LOCAL
RMS_STATUS;
MAP
FOREIGN_BUFFER : REF BLOCK [, BYTE];

Copy the current status of the cancel-control-0 bit in the LUB
(possibly set by RCTRL0) into the RAB, and clear it from the
LUB. The net effect of this is that if the bit is set in the
LUB, then the CANCTRL0 modifier will be applied to this write
operation only.

```
: 2843      4112 2 !-
: 2844
: 2845      4113
: 2846      4114  CCB [RABSV_CCO] = CCB [LUBSV_CCO];
: 2847      4115  CCB [LUBSV_CCO] = 0;
: 2848      4116
: 2849      4117
: 2850      4118  |+ Set the recordsize field in the RAB based on COUNT.
: 2851      4119  |+ Set the record address field in the RAB to the user buffer.
: 2852      4120  |+ Perform the PUT.
: 2853      4121  |+ If RMS returns a failure status, signal the error.
: 2854      4122
: 2855      4123
: 2856      4124  CCB [RABSW_RSZ] = .COUNT;
: 2857      4125  CCB [RABSB_RAC] = (IF .CCB [LUBSB_ORGAN] EQL LUBSK_ORG_INDEX THEN RABSC_KEY ELSE RABSC_SEQ);
: 2858      4126
: 2859      4127  IF .FOREIGN_BUFFER NEQA 0
: 2860      4128  THEN
: 2861      4129  |+ There is a foreign buffer. Point RABSL_UBF to it.
: 2862      4130  |-
: 2863      4131  |+ ELSE CCB [RABSL_RBF] = CCB [RABSL_UBF] = .FOREIGN_BUFFER [RABSL_UBF]
: 2864      4132  |+ ELSE CCB [RABSL_RBF] = .CCB [RABSL_UBF];
: 2865      4133
: 2866      4134  RMS_STATUS = SPUT (RAB = .CCB);
: 2867      4135
: 2868      4136  IF .RMS_STATUS EQL RMSS_CONTROLC
: 2869      4137  THEN
: 2870      4138  BASS$SIGNAL_CTRLC ();
: 2871      4139
: 2872      4140  IF NOT .RMS_STATUS
: 2873      4141  THEN
: 2874      4142  BEGIN
: 2875      4143
: 2876      4144
: 2877      4145  |+ We cannot call PUT_ERROR because we must restore UBF and USZ.
: 2878      4146
: 2879      4147
: 2880      4148
: 2881      4149  WHILE (.CCB [RABSL_STS] EQL RMSS_RSA) DO
: 2882      4150  BEGIN
: 2883      4151  SWAIT (RAB = .CCB);
: 2884      4152  SPUT (RAB = .CCB);
: 2885      4153  END;
: 2886      4154
: 2887      4155  END;
: 2888      4156
: 2889      4157
: 2890      4158  |+ Restore RABSL_UBF in case there was a foreign buffer.
: 2891      4159  |-
: 2892      4160  |+ CCB [RABSL_UBF] = .CCB [LUBSA_UBF];
: 2893      4161
: 2894      4162  |+ Point LUBSA_RBUF_PTR off to the buffer used by RMS.
: 2895      4163
: 2896      4164  |+ CCB [LUBSA_RBUF_ADDR] = .CCB [RABSL_UBF];
: 2897      4165
: 2898      4166  |+ Any error remaining (which will be an error other than Record Stream
: 2899      4167  Active, RSA) is fatal.
: 2 !-
```

```
2900      4169 2 IF ( NOT .CCB [RABSL_STS] ) THEN BASSSTOP_IO (BASSK_IOERR_REC);  
2901      4170 2  
2902      4171 2 RETURN;  
2903      4172 2  
2904      4173 1 END;                                ! End of BASSSREC_PSE
```

07		7E	AB	SE	04	C2 00000 BASSREC PSE::			
A0	AB	01		01	02	EF 00003	SUBL2	#4. SP	4055
		07		07	8E	FO 00009	EXTZV	#2, #1, -95((CB), -(SP))	4114
		A0	AB	04	8A	0000F	INSV	(SP)+ #7, #1, 7((CB))	
		22	AB	51	B0	00013	BICB2	#4, -96((CB))	4115
			03	AB	91	00017	MOVW	COUNT, 34((CB))	4124
				05	12	00018	CMPB	-60((CB)), #3	4125
				51	01	0001D	BNEQ	18	
					02	11	MOVL	#1, R1	
					51	04	BRB	2S	
				1E	AB	00022	CLRL	R1	
					51	90	MOVBL	R1, 30((CB))	
					50	00024	TSTL	FOREIGN_BUFFER	4127
				24	50	00028	BEQL	3S	
				28	AB	0002C	MOVL	36(FOREIGN BUFFER), R0	4132
					50	00030	MOVL	R0, 36((CB))	
				28	AB	00034	MOVL	R0, 40((CB))	
					05	11	BRB	4S	
				28	AB	0003A	MOVL	36((CB)), 40((CB))	4134
					5B	DD 0003F	PUSHL	CCB	4136
		00000000G	00		01	FB 00041	CALLS	#1, SYSSPUT	
			6E		50	DD 00048	MOVL	R0, RMS STATUS	
		00010651	8F		6E	D1 00048	CMPL	RMS_STATUS, #67153	4138
					07	12	BNEQ	5S	
		00000000G	00		00	FB 00054	CALLS	#0, BASS\$SIGNAL_CTRL_C	4140
			1E		6E	E8 00058	BLBS	RMS STATUS, 78	4142
		000182DA	8F	08	AB	D1 0005E	(MPL	8((CB)), #99034	4149
					14	12	BNEQ	7S	
					5B	DD 00068	PUSHL	CCB	4151
		00000000G	00		01	FB 0006A	CALLS	#1, SYSSHIFT	4152
					5B	DD 00071	PUSHL	CCB	
		00000000G	00		01	FB 00073	CALLS	#1, SYSSPUT	
					E2	11	BRB	6S	
		24	AB	9C	AB	DD 0007C	MOVL	-100((CB), 36((CB))	4149
		EC	AB	24	AB	DD 00081	MOVL	36((CB), -20((CB))	4160
		0A	08	AB	E8	00086	BLBS	8((CB)), 8S	4164
		7E		01	CE	0008A	MNEGL	#1, -(SP)	4170
		00000000G	00		01	FB 0008D	CALLS	#1, BASS\$STOP_IO	
			SE		04	CO 00094	ADDL2	#4, SP	
					05	00097	RSB		4173

; Routine Size: 152 bytes, Routine Base: _BASSCODE + 09DE

: 2905 4174 1

```

2907 4175 1 GLOBAL ROUTINE BASSSREC_PRE (
2908 4176 1   COUNT
2909 4177 1   FOREIGN_BUFFER
2910 4178 1 ) : JSB_PUT-NOVALUE =
2911 4179 1
2912 4180 1 ++
2913 4181 1   FUNCTIONAL DESCRIPTION:
2914 4182 1
2915 4183 1     Check for a foreign buffer and point to it if necessary.
2916 4184 1     Write one record. If successful then return; otherwise, signal a fatal
2917 4185 1     error.
2918 4186 1
2919 4187 1   FORMAL PARAMETERS:
2920 4188 1
2921 4189 1     COUNT.rl.v           No. of bytes to write
2922 4190 1     FOREIGN_BUFFER.rl.v  pointer to foreign (B or 0)
2923 4191 1
2924 4192 1   IMPLICIT INPUTS:
2925 4193 1
2926 4194 1     RABSL_UBF            for the foreign buffer (buffer pointer)
2927 4195 1     RABSW_USZ            for the foreign buffer (buffer size)
2928 4196 1
2929 4197 1   IMPLICIT OUTPUTS:
2930 4198 1
2931 4199 1     RABSW_RSZ             length of record to write (file buffer)
2932 4200 1     RABSL_RBF             pointer to file (B
2933 4201 1     LUBSL_LOG_RECNO      logical record number
2934 4202 1     RABSB_RAC             record access field
2935 4203 1
2936 4204 1   ROUTINE VALUE:
2937 4205 1
2938 4206 1     NONE
2939 4207 1
2940 4208 1   SIDE EFFECTS:
2941 4209 1
2942 4210 1     SIGNALS any RMS errors
2943 4211 1
2944 4212 1
2945 4213 2   BEGIN
2946 4214 2
2947 4215 2
2948 4216 2   EXTERNAL REGISTER
2949 4217 2     CCB : REF BLOCK [. BYTE];
2950 4218 2
2951 4219 2   LOCAL
2952 4220 2     RMS_STATUS;
2953 4221 2
2954 4222 2   MAP
2955 4223 2     FOREIGN_BUFFER : REF BLOCK [. BYTE];
2956 4224 2
2957 4225 2
2958 4226 2
2959 4227 2
2960 4228 2
2961 4229 2
2962 4230 2
2963 4231 2
2964 4232 2
2965 4233 2
2966 4234 2
2967 4235 2
2968 4236 2
2969 4237 2
2970 4238 2
2971 4239 2
2972 4240 2
2973 4241 2
2974 4242 2
2975 4243 2
2976 4244 2
2977 4245 2
2978 4246 2
2979 4247 2
2980 4248 2
2981 4249 2
2982 4250 2
2983 4251 2
2984 4252 2
2985 4253 2
2986 4254 2
2987 4255 2
2988 4256 2
2989 4257 2
2990 4258 2
2991 4259 2
2992 4260 2
2993 4261 2
2994 4262 2
2995 4263 2
2996 4264 2
2997 4265 2
2998 4266 2
2999 4267 2
3000 4268 2
3001 4269 2
3002 4270 2
3003 4271 2
3004 4272 2
3005 4273 2
3006 4274 2
3007 4275 2
3008 4276 2
3009 4277 2
3010 4278 2
3011 4279 2
3012 4280 2
3013 4281 2
3014 4282 2
3015 4283 2
3016 4284 2
3017 4285 2
3018 4286 2
3019 4287 2
3020 4288 2
3021 4289 2
3022 4290 2
3023 4291 2
3024 4292 2
3025 4293 2
3026 4294 2
3027 4295 2
3028 4296 2
3029 4297 2
3030 4298 2
3031 4299 2
3032 4300 2
3033 4301 2
3034 4302 2
3035 4303 2
3036 4304 2
3037 4305 2
3038 4306 2
3039 4307 2
3040 4308 2
3041 4309 2
3042 4310 2
3043 4311 2
3044 4312 2
3045 4313 2
3046 4314 2
3047 4315 2
3048 4316 2
3049 4317 2
3050 4318 2
3051 4319 2
3052 4320 2
3053 4321 2
3054 4322 2
3055 4323 2
3056 4324 2
3057 4325 2
3058 4326 2
3059 4327 2
3060 4328 2
3061 4329 2
3062 4330 2
3063 4331 2
3064 4332 2
3065 4333 2
3066 4334 2
3067 4335 2
3068 4336 2
3069 4337 2
3070 4338 2
3071 4339 2
3072 4340 2
3073 4341 2
3074 4342 2
3075 4343 2
3076 4344 2
3077 4345 2
3078 4346 2
3079 4347 2
3080 4348 2
3081 4349 2
3082 4350 2
3083 4351 2
3084 4352 2
3085 4353 2
3086 4354 2
3087 4355 2
3088 4356 2
3089 4357 2
3090 4358 2
3091 4359 2
3092 4360 2
3093 4361 2
3094 4362 2
3095 4363 2
3096 4364 2
3097 4365 2
3098 4366 2
3099 4367 2
3100 4368 2
3101 4369 2
3102 4370 2
3103 4371 2
3104 4372 2
3105 4373 2
3106 4374 2
3107 4375 2
3108 4376 2
3109 4377 2
3110 4378 2
3111 4379 2
3112 4380 2
3113 4381 2
3114 4382 2
3115 4383 2
3116 4384 2
3117 4385 2
3118 4386 2
3119 4387 2
3120 4388 2
3121 4389 2
3122 4390 2
3123 4391 2
3124 4392 2
3125 4393 2
3126 4394 2
3127 4395 2
3128 4396 2
3129 4397 2
3130 4398 2
3131 4399 2
3132 4400 2
3133 4401 2
3134 4402 2
3135 4403 2
3136 4404 2
3137 4405 2
3138 4406 2
3139 4407 2
3140 4408 2
3141 4409 2
3142 4410 2
3143 4411 2
3144 4412 2
3145 4413 2
3146 4414 2
3147 4415 2
3148 4416 2
3149 4417 2
3150 4418 2
3151 4419 2
3152 4420 2
3153 4421 2
3154 4422 2
3155 4423 2
3156 4424 2
3157 4425 2
3158 4426 2
3159 4427 2
3160 4428 2
3161 4429 2
3162 4430 2
3163 4431 2
3164 4432 2
3165 4433 2
3166 4434 2
3167 4435 2
3168 4436 2
3169 4437 2
3170 4438 2
3171 4439 2
3172 4440 2
3173 4441 2
3174 4442 2
3175 4443 2
3176 4444 2
3177 4445 2
3178 4446 2
3179 4447 2
3180 4448 2
3181 4449 2
3182 4450 2
3183 4451 2
3184 4452 2
3185 4453 2
3186 4454 2
3187 4455 2
3188 4456 2
3189 4457 2
3190 4458 2
3191 4459 2
3192 4460 2
3193 4461 2
3194 4462 2
3195 4463 2
3196 4464 2
3197 4465 2
3198 4466 2
3199 4467 2
3200 4468 2
3201 4469 2
3202 4470 2
3203 4471 2
3204 4472 2
3205 4473 2
3206 4474 2
3207 4475 2
3208 4476 2
3209 4477 2
3210 4478 2
3211 4479 2
3212 4480 2
3213 4481 2
3214 4482 2
3215 4483 2
3216 4484 2
3217 4485 2
3218 4486 2
3219 4487 2
3220 4488 2
3221 4489 2
3222 4490 2
3223 4491 2
3224 4492 2
3225 4493 2
3226 4494 2
3227 4495 2
3228 4496 2
3229 4497 2
3230 4498 2
3231 4499 2
3232 4500 2
3233 4501 2
3234 4502 2
3235 4503 2
3236 4504 2
3237 4505 2
3238 4506 2
3239 4507 2
3240 4508 2
3241 4509 2
3242 4510 2
3243 4511 2
3244 4512 2
3245 4513 2
3246 4514 2
3247 4515 2
3248 4516 2
3249 4517 2
3250 4518 2
3251 4519 2
3252 4520 2
3253 4521 2
3254 4522 2
3255 4523 2
3256 4524 2
3257 4525 2
3258 4526 2
3259 4527 2
3260 4528 2
3261 4529 2
3262 4530 2
3263 4531 2
3264 4532 2
3265 4533 2
3266 4534 2
3267 4535 2
3268 4536 2
3269 4537 2
3270 4538 2
3271 4539 2
3272 4540 2
3273 4541 2
3274 4542 2
3275 4543 2
3276 4544 2
3277 4545 2
3278 4546 2
3279 4547 2
3280 4548 2
3281 4549 2
3282 4550 2
3283 4551 2
3284 4552 2
3285 4553 2
3286 4554 2
3287 4555 2
3288 4556 2
3289 4557 2
3290 4558 2
3291 4559 2
3292 4560 2
3293 4561 2
3294 4562 2
3295 4563 2
3296 4564 2
3297 4565 2
3298 4566 2
3299 4567 2
3300 4568 2
3301 4569 2
3302 4570 2
3303 4571 2
3304 4572 2
3305 4573 2
3306 4574 2
3307 4575 2
3308 4576 2
3309 4577 2
3310 4578 2
3311 4579 2
3312 4580 2
3313 4581 2
3314 4582 2
3315 4583 2
3316 4584 2
3317 4585 2
3318 4586 2
3319 4587 2
3320 4588 2
3321 4589 2
3322 4590 2
3323 4591 2
3324 4592 2
3325 4593 2
3326 4594 2
3327 4595 2
3328 4596 2
3329 4597 2
3330 4598 2
3331 4599 2
3332 4600 2
3333 4601 2
3334 4602 2
3335 4603 2
3336 4604 2
3337 4605 2
3338 4606 2
3339 4607 2
3340 4608 2
3341 4609 2
3342 4610 2
3343 4611 2
3344 4612 2
3345 4613 2
3346 4614 2
3347 4615 2
3348 4616 2
3349 4617 2
3350 4618 2
3351 4619 2
3352 4620 2
3353 4621 2
3354 4622 2
3355 4623 2
3356 4624 2
3357 4625 2
3358 4626 2
3359 4627 2
3360 4628 2
3361 4629 2
3362 4630 2
3363 4631 2
3364 4632 2
3365 4633 2
3366 4634 2
3367 4635 2
3368 4636 2
3369 4637 2
3370 4638 2
3371 4639 2
3372 4640 2
3373 4641 2
3374 4642 2
3375 4643 2
3376 4644 2
3377 4645 2
3378 4646 2
3379 4647 2
3380 4648 2
3381 4649 2
3382 4650 2
3383 4651 2
3384 4652 2
3385 4653 2
3386 4654 2
3387 4655 2
3388 4656 2
3389 4657 2
3390 4658 2
3391 4659 2
3392 4660 2
3393 4661 2
3394 4662 2
3395 4663 2
3396 4664 2
3397 4665 2
3398 4666 2
3399 4667 2
3400 4668 2
3401 4669 2
3402 4670 2
3403 4671 2
3404 4672 2
3405 4673 2
3406 4674 2
3407 4675 2
3408 4676 2
3409 4677 2
3410 4678 2
3411 4679 2
3412 4680 2
3413 4681 2
3414 4682 2
3415 4683 2
3416 4684 2
3417 4685 2
3418 4686 2
3419 4687 2
3420 4688 2
3421 4689 2
3422 4690 2
3423 4691 2
3424 4692 2
3425 4693 2
3426 4694 2
3427 4695 2
3428 4696 2
3429 4697 2
3430 4698 2
3431 4699 2
3432 4700 2
3433 4701 2
3434 4702 2
3435 4703 2
3436 4704 2
3437 4705 2
3438 4706 2
3439 4707 2
3440 4708 2
3441 4709 2
3442 4710 2
3443 4711 2
3444 4712 2
3445 4713 2
3446 4714 2
3447 4715 2
3448 4716 2
3449 4717 2
3450 4718 2
3451 4719 2
3452 4720 2
3453 4721 2
3454 4722 2
3455 4723 2
3456 4724 2
3457 4725 2
3458 4726 2
3459 4727 2
3460 4728 2
3461 4729 2
3462 4730 2
3463 4731 2
3464 4732 2
3465 4733 2
3466 4734 2
3467 4735 2
3468 4736 2
3469 4737 2
3470 4738 2
3471 4739 2
3472 4740 2
3473 4741 2
3474 4742 2
3475 4743 2
3476 4744 2
3477 4745 2
3478 4746 2
3479 4747 2
3480 4748 2
3481 4749 2
3482 4750 2
3483 4751 2
3484 4752 2
3485 4753 2
3486 4754 2
3487 4755 2
3488 4756 2
3489 4757 2
3490 4758 2
3491 4759 2
3492 4760 2
3493 4761 2
3494 4762 2
3495 4763 2
3496 4764 2
3497 4765 2
3498 4766 2
3499 4767 2
3500 4768 2
3501 4769 2
3502 4770 2
3503 4771 2
3504 4772 2
3505 4773 2
3506 4774 2
3507 4775 2
3508 4776 2
3509 4777 2
3510 4778 2
3511 4779 2
3512 4780 2
3513 4781 2
3514 4782 2
3515 4783 2
3516 4784 2
3517 4785 2
3518 4786 2
3519 4787 2
3520 4788 2
3521 4789 2
3522 4790 2
3523 4791 2
3524 4792 2
3525 4793 2
3526 4794 2
3527 4795 2
3528 4796 2
3529 4797 2
3530 4798 2
3531 4799 2
3532 4800 2
3533 4801 2
3534 4802 2
3535 4803 2
3536 4804 2
3537 4805 2
3538 4806 2
3539 4807 2
3540 4808 2
3541 4809 2
3542 4810 2
3543 4811 2
3544 4812 2
3545 4813 2
3546 4814 2
3547 4815 2
3548 4816 2
3549 4817 2
3550 4818 2
3551 4819 2
3552 4820 2
3553 4821 2
3554 4822 2
3555 4823 2
3556 4824 2
3557 4825 2
3558 4826 2
3559 4827 2
3560 4828 2
3561 4829 2
3562 4830 2
3563 4831 2
3564 4832 2
3565 4833 2
3566 4834 2
3567 4835 2
3568 4836 2
3569 4837 2
3570 4838 2
3571 4839 2
3572 4840 2
3573 4841 2
3574 4842 2
3575 4843 2
3576 4844 2
3577 4845 2
3578 4846 2
3579 4847 2
3580 4848 2
3581 4849 2
3582 4850 2
3583 4851 2
3584 4852 2
3585 4853 2
3586 4854 2
3587 4855 2
3588 4856 2
3589 4857 2
3590 4858 2
3591 4859 2
3592 4860 2
3593 4861 2
3594 4862 2
3595 4863 2
3596 4864 2
3597 4865 2
3598 4866 2
3599 4867 2
3600 4868 2
3601 4869 2
3602 4870 2
3603 4871 2
3604 4872 2
3605 4873 2
3606 4874 2
3607 4875 2
3608 4876 2
3609 4877 2
3610 4878 2
3611 4879 2
3612 4880 2
3613 4881 2
3614 4882 2
3615 4883 2
3616 4884 2
3617 4885 2
3618 4886 2
3619 4887 2
3620 4888 2
3621 4889 2
3622 4890 2
3623 4891 2
3624 4892 2
3625 4893 2
3626 4894 2
3627 4895 2
3628 4896 2
3629 4897 2
3630 4898 2
3631 4899 2
3632 4900 2
3633 4901 2
3634 4902 2
3635 4903 2
3636 4904 2
3637 4905 2
3638 4906 2
3639 4907 2
3640 4908 2
3641 4909 2
3642 4910 2
3643 4911 2
3644 4912 2
3645 4913 2
3646 4914 2
3647 4915 2
3648 4916 2
3649 4917 2
3650 4918 2
3651 4919 2
3652 4920 2
3653 4921 2
3654 4922 2
3655 4923 2
3656 4924 2
3657 4925 2
3658 4926 2
3659 4927 2
3660 4928 2
3661 4929 2
3662 4930 2
3663 4931 2
3664 4932 2
3665 4933 2
3666 4934 2
3667 4935 2
3668 4936 2
3669 4937 2
3670 4938 2
3671 4939 2
3672 4940 2
3673 4941 2
3674 4942 2
3675 4943 2
3676 4944 2
3677 4945 2
3678 4946 2
3679 4947 2
3680 4948 2
3681 4949 2
3682 4950 2
3683 4951 2
3684 4952 2
3685 4953 2
3686 4954 2
3687 4955 2
3688 4956 2
3689 4957 2
3690 4958 2
3691 4959 2
3692 4960 2
3693 4961 2
3694 4962 2
3695 4963 2
3696 4964 2
3697 4965 2
3698 4966 2
3699 4967 2
3700 4968 2
3701 4969 2
3702 4970 2
3703 4971 2
3704 4972 2
3705 4973 2
3706 4974 2
3707 4975 2
3708 4976 2
3709 4977 2
3710 4978 2
3711 4979 2
3712 4980 2
3713 4981 2
3714 4982 2
3715 4983 2
3716 4984 2
3717 4985 2
3718 4986 2
3719 4987 2
3720 4988 2
3721 4989 2
3722 4990 2
3723 4991 2
3724 4992 2
3725 4993 2
3726 4994 2
3727 4995 2
3728 4996 2
3729 4997 2
3730 4998 2
3731 4999 2
3732 5000 2
3733 5001 2
3734 5002 2
3735 5003 2
3736 5004 2
3737 5005 2
3738 5006 2
3739 5007 2
3740 5008 2
3741 5009 2
3742 5010 2
3743 5011 2
3744 5012 2
3745 5013 2
3746 5014 2
3747 5015 2
3748 5016 2
3749 5017 2
3750 5018 2
3751 5019 2
3752 5020 2
3753 5021 2
3754 5022 2
3755 5023 2
3756 5024 2
3757 5025 2
3758 5026 2
3759 5027 2
3760 5028 2
3761 5029 2
3762 5030 2
3763 5031 2
3764 5032 2
3765 5033 2
3766 5034 2
3767 5035 2
3768 5036 2
3769 5037 2
3770 5038 2
3771 5039 2
3772 5040 2
3773 5041 2
3774 5042 2
3775 5043 2
3776 5044 2
3777 5045 2
3778 5046 2
3779 5047 2
3780 5048 2
3781 5049 2
3782 5050 2
3783 5051 2
3784 5052 2
3785 5053 2
3786 5054 2
3787 5055 2
3788 5056 2
3789 5057 2
3790 5058 2
3791 5059 2
3792 5060 2
3793 5061 2
3794 5062 2
3795 5063 2
3796 5064 2
3797 5065 2
3798 5066 2
3799 5067 2
3800 5068 2
3801 5069 2
3802 5070 2
3803 5071 2
3804 5072 2
3805 5073 2
3806 5074 2
3807 5075 2
3808 5076 2
3809 5077 2
3810 5078 2
3811 5079 2
3812 5080 2
3813 5081 2
3814 5082 2
3815 5083 2
3816 5084 2
3817 5085 2
3818 5086 2
3819 5087 2
3820 5088 2
3821 5089 2
3822 5090 2
3823 5091 2
3824 5092 2
3825 5093 2
3826 5094 2
3827 5095 2
3828 5096 2
3829 5097 2
3830 5098 2
3831 5099 2
3
```

```
2964      4232 2 CCB [RABSB_RAC] = RABSC_KEY;  
2965      4233 2 IF .FOREIGN_BUFFER NEQA 0  
2966      4234 2 THEN  
2967      4235 2  
2968      4236 2  
2969      4237 2      * There is a foreign buffer. Point off to the buffer but don't do the  
2970      4238 2      size. A PUT with count would not work right, so the size is passed in.  
2971      4239 2  
2972      4240 2      -  
2973      4241 2      ELSE CCB [RABSL_UBF] = CCB [RABSL_RBF] = .FOREIGN_BUFFER [RABSL_UBF]  
2974      4242 2      ELSE CCB [RABSL_RBF] = .CCB [RABSL_UBF];  
2975      4243 2  
2976      4244 2 RMS_STATUS = $PUT (RAB = .CCB);  
2977      4245 2  
2978      4246 2 IF .RMS_STATUS EQL RMSS_CONTROLC  
2979      4247 2 THEN  
2980      4248 2      BAS$SIGNAL_CTRLC ();  
2981      4249 2  
2982      4250 2 IF NOT .RMS_STATUS  
2983      4251 2 THEN  
2984      4252 2 BEGIN  
2985      4253 2  
2986      4254 2      * We cannot call GET_ERROR because we must restore UBF and USZ.  
2987      4255 2  
2988      4256 2  
2989      4257 2      WHILE (.CCB [RABSL_STS] EQL RMSS_RSA) DO  
2990      4258 2      BEGIN  
2991      4259 2      SWAIT (RAB = .CCB);  
2992      4260 2      $PUT (RAB = .CCB);  
2993      4261 2      END;  
2994      4262 2  
2995      4263 2  
2996      4264 2  
2997      4265 2  
2998      4266 2      * Restore RABSL_UBF in case there was a foreign buffer.  
2999      4267 2  
3000      4268 2      CCB [RABSL_UBF] = .CCB [LUBSA_UBF];  
3001      4269 2  
3002      4270 2      Point LUBSA_BUF_PTR off to the buffer used by RMS.  
3003      4271 2  
3004      4272 2      CCB [LUBSA_RBUF_ADR] = .CCB [RABSL_UBF];  
3005      4273 2  
3006      4274 2  
3007      4275 2      Any error remaining (which will be an error other than Record Stream  
3008      4276 2      Active, RSA) is fatal.  
3009      4277 2  
3010      4278 2      IF ( NOT .CCB [RABSL_STS]) THEN BAS$STOP_IO (BASSK_IOERR_REC);  
3011      4279 2  
3012      4280 2      RETURN;  
3013      4281 2      END;                      ! End of BASS$REC_PRE
```

22	AB		51	B0 00003	MOVW	COUNT 34((CB))	: 4231
1E	AB		01	90 00007	MOVB	#1 30((CB))	: 4232
			50	D5 00008	TSTL	FOREIGN_BUFFER	: 4234
			0E	15 0000D	BEQL	1\$	
28	50	24	A0	D0 0000F	MOVL	36(FOREIGN_BUFFER), R0	: 4240
24	AB		50	D0 00013	MOVL	R0, 40((CB))	
			50	D0 00017	MOVL	R0, 36((CB))	
28	AB	24	05	11 0001B	BRB	2\$	
			AB	D0 0001D	MOVL	36((CB)), 40((CB))	: 4242
00000000G	00		5B	DD 00022	PUSHL	((CB))	: 4244
	6E		01	FB 00024	CALLS	#1, SYSSPUT	
00010651	8F		50	D0 0002B	MOVL	R0, RMS STATUS	
			6E	D1 0002E	CMPL	RMS_STATUS, #67153	: 4246
00000000G	00		07	12 00035	BNEQ	3\$	
	1E		00	FB 00037	CALLS	#0, BASS\$SIGNAL_CTRLC	: 4248
000182DA	8F	08	6E	E8 0003E	BLBS	RMS_STATUS, 5\$: 4250
			AB	D1 00041	CMPL	8((CB)), #99034	: 4257
			14	12 00049	BNEQ	5\$	
00000000G	00		5B	DD 0004B	PUSHL	((CB))	: 4259
			01	FB 0004D	CALLS	#1, SYSSWAIT	
00000000G	00		5B	DD 00054	PUSHL	((CB))	: 4260
			01	FB 00056	CALLS	#1, SYSSPUT	
			E2	11 0005D	BRB	4\$	
24	AB	9C	AB	D0 0005F	MOVL	-100((CB)), 36((CB))	: 4268
EC	AB	24	AB	D0 00064	MOVL	36((CB)), -20((CB))	: 4272
	0A	08	AB	E8 00069	BLBS	8((CB)), 6\$: 4278
00000000G	00		01	CE 0006D	MNEG	#1, -(SP)	
	7E		01	FB 00070	CALLS	#1, BASS\$STOP_10	
	5E		04	C0 00077	ADDL2	#4, SP	
			05	0007A	RSB		: 4281

: Routine Size: 123 bytes, Routine Base: _BASS\$CODE + 0A76

: 3014 4282 1

J 10
16-Sep-1984 01:01:12 VAX-11 Bliss-32 v4.0-742
14-Sep-1984 11:56:35 [BASRTL.SRC]BASREC[PRO.B32;1]

```

3016 4283 1 GLOBAL ROUTINE BASSSREC_FSE (           ! FIND (sequential) a record
3017 4284 1   LOCK_FLAGS
3018 4285 1   ) : JSB_REC2 NOVALUE =
3019 4286 1
3020 4287 1 ++
3021 4288 1   FUNCTIONAL DESCRIPTION:
3022 4289 1
3023 4290 1     Find next record. If successful then return; otherwise, signal a fatal
3024 4291 1     error.
3025 4292 1
3026 4293 1   FORMAL PARAMETERS:
3027 4294 1
3028 4295 1     LOCK_FLAGS.rlu.v      bits to set in the RAB ROP to control manual
3029 4296 1                           record locking (0 if none)
3030 4297 1
3031 4298 1   IMPLICIT INPUTS:
3032 4299 1     NONE
3033 4300 1
3034 4301 1   IMPLICIT OUTPUTS:
3035 4302 1     RABSB_RAC          record access field
3036 4303 1
3037 4304 1   ROUTINE VALUE:
3038 4305 1     NONE
3039 4306 1
3040 4307 1   SIDE EFFECTS:
3041 4308 1     Finds next record in file on this logical unit.
3042 4309 1     SIGNALS any RMS errors
3043 4310 1
3044 4311 1
3045 4312 1
3046 4313 1
3047 4314 1
3048 4315 1
3049 4316 2   BEGIN
3050 4317 2
3051 4318 2   EXTERNAL REGISTER
3052 4319 2     CCB : REF BLOCK [, BYTE];
3053 4320 2
3054 4321 2
3055 4322 2   LOCAL
3056 4323 2     RMS_STATUS;
3057 4324 2
3058 4325 2
3059 4326 2
3060 4327 2
3061 4328 2
3062 4329 2
3063 4330 2
3064 4331 2
3065 4332 2
3066 4333 2
3067 4334 2
3068 4335 2
3069 4336 2
3070 4337 2
3071 4338 2
3072 4339 2
3073 4340 2
3074 4341 2
3075 4342 2
3076 4343 2
3077 4344 2
3078 4345 2
3079 4346 2
3080 4347 2
3081 4348 2
3082 4349 2
3083 4350 2
3084 4351 2
3085 4352 2
3086 4353 2
3087 4354 2
3088 4355 2
3089 4356 2
3090 4357 2
3091 4358 2
3092 4359 2
3093 4360 2
3094 4361 2
3095 4362 2
3096 4363 2
3097 4364 2
3098 4365 2
3099 4366 2
3100 4367 2
3101 4368 2
3102 4369 2
3103 4370 2
3104 4371 2
3105 4372 2
3106 4373 2
3107 4374 2
3108 4375 2
3109 4376 2
3110 4377 2
3111 4378 2
3112 4379 2
3113 4380 2
3114 4381 2
3115 4382 2
3116 4383 2
3117 4384 2
3118 4385 2
3119 4386 2
3120 4387 2
3121 4388 2
3122 4389 2
3123 4390 2
3124 4391 2
3125 4392 2
3126 4393 2
3127 4394 2
3128 4395 2
3129 4396 2
3130 4397 2
3131 4398 2
3132 4399 2
3133 4400 2
3134 4401 2
3135 4402 2
3136 4403 2
3137 4404 2
3138 4405 2
3139 4406 2
3140 4407 2
3141 4408 2
3142 4409 2
3143 4410 2
3144 4411 2
3145 4412 2
3146 4413 2
3147 4414 2
3148 4415 2
3149 4416 2
3150 4417 2
3151 4418 2
3152 4419 2
3153 4420 2
3154 4421 2
3155 4422 2
3156 4423 2
3157 4424 2
3158 4425 2
3159 4426 2
3160 4427 2
3161 4428 2
3162 4429 2
3163 4430 2
3164 4431 2
3165 4432 2
3166 4433 2
3167 4434 2
3168 4435 2
3169 4436 2
3170 4437 2
3171 4438 2
3172 4439 2
3173 4440 2
3174 4441 2
3175 4442 2
3176 4443 2
3177 4444 2
3178 4445 2
3179 4446 2
3180 4447 2
3181 4448 2
3182 4449 2
3183 4450 2
3184 4451 2
3185 4452 2
3186 4453 2
3187 4454 2
3188 4455 2
3189 4456 2
3190 4457 2
3191 4458 2
3192 4459 2
3193 4460 2
3194 4461 2
3195 4462 2
3196 4463 2
3197 4464 2
3198 4465 2
3199 4466 2
3200 4467 2
3201 4468 2
3202 4469 2
3203 4470 2
3204 4471 2
3205 4472 2
3206 4473 2
3207 4474 2
3208 4475 2
3209 4476 2
3210 4477 2
3211 4478 2
3212 4479 2
3213 4480 2
3214 4481 2
3215 4482 2
3216 4483 2
3217 4484 2
3218 4485 2
3219 4486 2
3220 4487 2
3221 4488 2
3222 4489 2
3223 4490 2
3224 4491 2
3225 4492 2
3226 4493 2
3227 4494 2
3228 4495 2
3229 4496 2
3230 4497 2
3231 4498 2
3232 4499 2
3233 4500 2
3234 4501 2
3235 4502 2
3236 4503 2
3237 4504 2
3238 4505 2
3239 4506 2
3240 4507 2
3241 4508 2
3242 4509 2
3243 4510 2
3244 4511 2
3245 4512 2
3246 4513 2
3247 4514 2
3248 4515 2
3249 4516 2
3250 4517 2
3251 4518 2
3252 4519 2
3253 4520 2
3254 4521 2
3255 4522 2
3256 4523 2
3257 4524 2
3258 4525 2
3259 4526 2
3260 4527 2
3261 4528 2
3262 4529 2
3263 4530 2
3264 4531 2
3265 4532 2
3266 4533 2
3267 4534 2
3268 4535 2
3269 4536 2
3270 4537 2
3271 4538 2
3272 4539 2
3273 4540 2
3274 4541 2
3275 4542 2
3276 4543 2
3277 4544 2
3278 4545 2
3279 4546 2
3280 4547 2
3281 4548 2
3282 4549 2
3283 4550 2
3284 4551 2
3285 4552 2
3286 4553 2
3287 4554 2
3288 4555 2
3289 4556 2
3290 4557 2
3291 4558 2
3292 4559 2
3293 4560 2
3294 4561 2
3295 4562 2
3296 4563 2
3297 4564 2
3298 4565 2
3299 4566 2
3300 4567 2
3301 4568 2
3302 4569 2
3303 4570 2
3304 4571 2
3305 4572 2
3306 4573 2
3307 4574 2
3308 4575 2
3309 4576 2
3310 4577 2
3311 4578 2
3312 4579 2
3313 4580 2
3314 4581 2
3315 4582 2
3316 4583 2
3317 4584 2
3318 4585 2
3319 4586 2
3320 4587 2
3321 4588 2
3322 4589 2
3323 4590 2
3324 4591 2
3325 4592 2
3326 4593 2
3327 4594 2
3328 4595 2
3329 4596 2
3330 4597 2
3331 4598 2
3332 4599 2
3333 4600 2
3334 4601 2
3335 4602 2
3336 4603 2
3337 4604 2
3338 4605 2
3339 4606 2
3340 4607 2
3341 4608 2
3342 4609 2
3343 4610 2
3344 4611 2
3345 4612 2
3346 4613 2
3347 4614 2
3348 4615 2
3349 4616 2
3350 4617 2
3351 4618 2
3352 4619 2
3353 4620 2
3354 4621 2
3355 4622 2
3356 4623 2
3357 4624 2
3358 4625 2
3359 4626 2
3360 4627 2
3361 4628 2
3362 4629 2
3363 4630 2
3364 4631 2
3365 4632 2
3366 4633 2
3367 4634 2
3368 4635 2
3369 4636 2
3370 4637 2
3371 4638 2
3372 4639 2
3373 4640 2
3374 4641 2
3375 4642 2
3376 4643 2
3377 4644 2
3378 4645 2
3379 4646 2
3380 4647 2
3381 4648 2
3382 4649 2
3383 4650 2
3384 4651 2
3385 4652 2
3386 4653 2
3387 4654 2
3388 4655 2
3389 4656 2
3390 4657 2
3391 4658 2
3392 4659 2
3393 4660 2
3394 4661 2
3395 4662 2
3396 4663 2
3397 4664 2
3398 4665 2
3399 4666 2
3400 4667 2
3401 4668 2
3402 4669 2
3403 4670 2
3404 4671 2
3405 4672 2
3406 4673 2
3407 4674 2
3408 4675 2
3409 4676 2
3410 4677 2
3411 4678 2
3412 4679 2
3413 4680 2
3414 4681 2
3415 4682 2
3416 4683 2
3417 4684 2
3418 4685 2
3419 4686 2
3420 4687 2
3421 4688 2
3422 4689 2
3423 4690 2
3424 4691 2
3425 4692 2
3426 4693 2
3427 4694 2
3428 4695 2
3429 4696 2
3430 4697 2
3431 4698 2
3432 4699 2
3433 4700 2
3434 4701 2
3435 4702 2
3436 4703 2
3437 4704 2
3438 4705 2
3439 4706 2
3440 4707 2
3441 4708 2
3442 4709 2
3443 4710 2
3444 4711 2
3445 4712 2
3446 4713 2
3447 4714 2
3448 4715 2
3449 4716 2
3450 4717 2
3451 4718 2
3452 4719 2
3453 4720 2
3454 4721 2
3455 4722 2
3456 4723 2
3457 4724 2
3458 4725 2
3459 4726 2
3460 4727 2
3461 4728 2
3462 4729 2
3463 4730 2
3464 4731 2
3465 4732 2
3466 4733 2
3467 4734 2
3468 4735 2
3469 4736 2
3470 4737 2
3471 4738 2
3472 4739 2
3473 4740 2
3474 4741 2
3475 4742 2
3476 4743 2
3477 4744 2
3478 4745 2
3479 4746 2
3480 4747 2
3481 4748 2
3482 4749 2
3483 4750 2
3484 4751 2
3485 4752 2
3486 4753 2
3487 4754 2
3488 4755 2
3489 4756 2
3490 4757 2
3491 4758 2
3492 4759 2
3493 4760 2
3494 4761 2
3495 4762 2
3496 4763 2
3497 4764 2
3498 4765 2
3499 4766 2
3500 4767 2
3501 4768 2
3502 4769 2
3503 4770 2
3504 4771 2
3505 4772 2
3506 4773 2
3507 4774 2
3508 4775 2
3509 4776 2
3510 4777 2
3511 4778 2
3512 4779 2
3513 4780 2
3514 4781 2
3515 4782 2
3516 4783 2
3517 4784 2
3518 4785 2
3519 4786 2
3520 4787 2
3521 4788 2
3522 4789 2
3523 4790 2
3524 4791 2
3525 4792 2
3526 4793 2
3527 4794 2
3528 4795 2
3529 4796 2
3530 4797 2
3531 4798 2
3532 4799 2
3533 4800 2
3534 4801 2
3535 4802 2
3536 4803 2
3537 4804 2
3538 4805 2
3539 4806 2
3540 4807 2
3541 4808 2
3542 4809 2
3543 4810 2
3544 4811 2
3545 4812 2
3546 4813 2
3547 4814 2
3548 4815 2
3549 4816 2
3550 4817 2
3551 4818 2
3552 4819 2
3553 4820 2
3554 4821 2
3555 4822 2
3556 4823 2
3557 4824 2
3558 4825 2
3559 4826 2
3560 4827 2
3561 4828 2
3562 4829 2
3563 4830 2
3564 4831 2
3565 4832 2
3566 4833 2
3567 4834 2
3568 4835 2
3569 4836 2
3570 4837 2
3571 4838 2
3572 4839 2
3573 4840 2
3574 4841 2
3575 4842 2
3576 4843 2
3577 4844 2
3578 4845 2
3579 4846 2
3580 4847 2
3581 4848 2
3582 4849 2
3583 4850 2
3584 4851 2
3585 4852 2
3586 4853 2
3587 4854 2
3588 4855 2
3589 4856 2
3590 4857 2
3591 4858 2
3592 4859 2
3593 4860 2
3594 4861 2
3595 4862 2
3596 4863 2
3597 4864 2
3598 4865 2
3599 4866 2
3600 4867 2
3601 4868 2
3602 4869 2
3603 4870 2
3604 4871 2
3605 4872 2
3606 4873 2
3607 4874 2
3608 4875 2
3609 4876 2
3610 4877 2
3611 4878 2
3612 4879 2
3613 4880 2
3614 4881 2
3615 4882 2
3616 4883 2
3617 4884 2
3618 4885 2
3619 4886 2
3620 4887 2
3621 4888 2
3622 4889 2
3623 4890 2
3624 4891 2
3625 4892 2
3626 4893 2
3627 4894 2
3628 4895 2
3629 4896 2
3630 4897 2
3631 4898 2
3632 4899 2
3633 4900 2
3634 4901 2
3635 4902 2
3636 4903 2
3637 4904 2
3638 4905 2
3639 4906 2
3640 4907 2
3641 4908 2
3642 4909 2
3643 4910 2
3644 4911 2
3645 4912 2
3646 4913 2
3647 4914 2
3648 4915 2
3649 4916 2
3650 4917 2
3651 4918 2
3652 4919 2
3653 4920 2
3654 4921 2
3655 4922 2
3656 4923 2
3657 4924 2
3658 4925 2
3659 4926 2
3660 4927 2
3661 4928 2
3662 4929 2
3663 4930 2
3664 4931 2
3665 4932 2
3666 4933 2
3667 4934 2
3668 4935 2
3669 4936 2
3670 4937 2
3671 4938 2
3672 4939 2
3673 4940 2
3674 4941 2
3675 4942 2
3676 4943 2
3677 4944 2
3678 4945 2
3679 4946 2
3680 4947 2
3681 4948 2
3682 4949 2
3683 4950 2
3684 4951 2
3685 4952 2
3686 4953 2
3687 4954 2
3688 4955 2
3689 4956 2
3690 4957 2
3691 4958 2
3692 4959 2
3693 4960 2
3694 4961 2
3695 4962 2
3696 4963 2
3697 4964 2
3698 4965 2
3699 4966 2
3700 4967 2
3701 4968 2
3702 4969 2
3703 4970 2
3704 4971 2
3705 4972 2
3706 4973 2
3707 4974 2
3708 4975 2
3709 4976 2
3710 4977 2
3711 4978 2
3712 4979 2
3713 4980 2
3714 4981 2
3715 4982 2
3716 4983 2
3717 4984 2
3718 4985 2
3719 4986 2
3720 4987 2
3721 4988 2
3722 4989 2
3723 4990 2
3724 4991 2
3725 4992 2
3726 4993 2
3727 4994 2
3728 4995 2
3729 4996 2
3730 4997 2
3731 4998 2
3732 4999 2
3733 5000 2
3734 5001 2
3735 5002 2
3736 5003 2
3737 5004 2
3738 5005 2
3739 5006 2
3740 5007 2
3741 5008 2
3742 5009 2
3743 5010 2
3744 5011 2
3745 5012 2
3746 5013 2
3747 5014 2
3748 5015 2
3749 5016 2
3750 5017 2
3751 5018 2
3752 5019 2
3753 5020 2
3754 5021 2
3755 5022 2
3756 5023 2
3757 5024 2
3758 5025 2
3759 5026 2
3760 5027 2
3761 5028 2
3762 5029 2
3763 5030 2
3764 5031 2
3765 5032 2
3766 5033 2
3767 5034 2
3768 5035 2
3769 5036 2
3770 5037 2
3771 5038 2
3772 5039 2
3773 5040 2
3774 5041 2
3775 5042 2
3776 5043 2
3777 5044 2
3778 5045 2
3779 5046 2
3780 5047 2
3781 5048 2
3782 5049 2
3783 5050 2
3784 5051 2
3785 5052 2
3786 5053 2
3787 5054 2
3788 5055 2
3789 5056 2
3790 5057 2
3791 5058 2
3792 5059 2
3793 5060 2
3794 5061 2
3795 5062 2
3796 5063 2
3797 5064 2
3798 5065 2
3799 5066 2
3800 5067 2
3801 5068 2
3802 5069 2
3803 5070 2
3804 5071 2
3805 5072 2
3806 5073 2
3807 5074 2
3808 5075 2
3809 5076 2
3810 5077 2
3811 5078 2
3812 5079 2
3813 5080 2
3814 5081 2
3815 5082 2
3816 5083 2
3817 5084 2
3818 5085 2
3819 5086 2
3820 5087 2
3821 5088 2
3822 5089 2
3823 5090 2
3824 5091 2
3825 5092 2
3826 5093 2
3827 5094 2
3828 5095 2
3829 5096 2
3830 5097 2
3831 5098 2
3832 5099 2
3833 5100 2
3834 5101 2
3835 5102 2
3836 5103 2
3837 5104 2
3838 5105 2
3839 5106 2
3840 5107 2
3841 5108 2
3842 5109 2
3843 5110 2
3844 5111 2
3845 5112 2
3846 5113 2
3847 5114 2
3848 5115 2
3849 5116 2
3850 5117 2
3851 5118 2
3852 5119 2
3853 5120 2
3854 5121 2
3855 5122 2
3856 5123 2
3857 5124 2
3858 5125 2
3859 5126 2
3860 5127 2
3861 5128 2
3862 5129 2
3863 5130 2
3864 5131 2
3865 5132 2
3866 5133 2
3867 5134 2
3868 5135 2
3869 5136 2
3870 5137 2
3871 5138 2
3872 5139 2
3873 5140 2
3874 5141 2
3875 5142 2
3876 5143 2
3877 5144 2
3878 5145 2
3879 5146 2
3880 5147 2
3881 5148 2
3882 5149 2
3883 5150 2
3884 5151 2
3885 5152 2
3886 5153 2
3887 5154 2
3888 5155 2
3889 5156 2
3890 5157 2
3891 5158 2
3892 5159 2
3893 5160 2
3894 5161 2
3895 5162 2
3896 5163 2
3897 5164 2
3898 5165 2
3899 5166 2
3900 5167 2
3901 5168 2
3902 5169 2
3903 5170 2
3904 5171 2
3905 5172 2
3906 5173 2
3907 5174 2
3908 5175 2
3909 5176 2
3910 5177 2
3911 5178 2
3912 5179 2
3913 5180 2
3914 5181 2
3915 5182 2
3916 5183 2
3917 5184 2
3918 5185 2
3919 5186 2
3920 5187 2
3921 5188 2
3922 5189 2
3923 5190 2
3924 5191 2
3925 5192 2
3926 5193 2
3927 5194 2
3928 5195 2
3929 5196 2
3930 5197 2
3931 5198 2
3932 5199 2
3933 5200 2
3934 5201 2
3935 5202 2
3936 5203 2
3937 5204 2
3938 5205 2
3939 5206 2
3940 5207 2
3941 5208 2
3942 5209 2
3943 5210 2
3944 5211 2
3945 5212 2
3946 521
```

```

3073    4340 2
3074    4341 2
3075    4342 2
3076    4343 2
3077    4344 2
3078    4345 2
3079    4346 2
3080    4347 2
3081    4348 2
3082    4349 2
3083    4350 2
3084    4351 2
3085    4352 2
3086    4353 2
3087    4354 2
3088    4355 2
3089    4356 2
3090    4357 1

RMS_STATUS = SFIND (RAB = .CCB);

!+
! Turn off bits so that subsequent operations will not inherit them.

CCB [RABSL_ROP] = .CCB [RABSL_ROP] XOR .LOCK_FLAGS;

!-
! signal if the FIND failed.

IF NOT .RMS_STATUS
THEN
    BASS$STOP_IO (BASS$IOERR_REC);

RETURN;
END;

```

! End of BASS\$REC_FSE

.EXTRN SYSSFIND

		52	DD 00000 BASS\$REC FSE::		
			PUSHL	R2	4283
			MOVL	R0, R2	4329
			CLRB	30(CCB)	4335
		52	BISL2	LOCK_FLAGS, 4(CCB)	4341
		AB	PUSHL	CCB	
		52	CALLS	#1. SYSSFIND	
04	AB	58	XROR2	LOCK_FLAGS, 4(CCB)	4347
		01	BLBS	RMS_STATUS, 1\$	4352
00000000G	00	52	MNEGL	#1. -(SP)	4354
		01	CALLS	#1. BASS\$STOP_IO	
04	AB	52	POPR	#^M<R2>	
		0A	RSB		
00000000G	00	01			
		04			
		BA 00026	1\$:		
		05 00028			

; Routine Size: 41 bytes. Routine Base: _BASS\$CODE + 0AF1

```
3092      4358 1 GLOBAL ROUTINE BASSREC_FRFA (           ! FIND (by RFA) a record
3093          4359 1   LOCK_FLAGS
3094          4360 1   ) : JSB_REC2 NOVALUE =
3095          4361 1
3096          4362 1
3097          4363 1
3098          4364 1
3099          4365 1
3100          4366 1
3101          4367 1
3102          4368 1
3103          4369 1
3104          4370 1
3105          4371 1
3106          4372 1
3107          4373 1
3108          4374 1
3109          4375 1
3110          4376 1
3111          4377 1
3112          4378 1
3113          4379 1
3114          4380 1
3115          4381 1
3116          4382 1
3117          4383 1
3118          4384 1
3119          4385 1
3120          4386 1
3121          4387 1
3122          4388 1
3123          4389 1
3124          4390 1
3125          4391 2
3126          4392 2
3127          4393 2
3128          4394 2
3129          4395 2
3130          4396 2
3131          4397 2
3132          4398 2
3133          4399 2
3134          4400 2
3135          4401 2
3136          4402 2
3137          4403 2
3138          4404 2
3139          4405 2
3140          4406 2
3141          4407 2
3142          4408 2
3143          4409 2
3144          4410 2
3145          4411 2
3146          4412 2
3147          4413 2
3148          4414 2
        1   FUNCTIONAL DESCRIPTION:
        1     Find record by RFA stored in the RAB. If successful then return; otherwise, signal a fatal
        1     error.
        1
        1   FORMAL PARAMETERS:
        1
        1     LOCK_FLAGS.rlu.v      bits to set in the RAB ROP to control manual
        1                           record locking (0 if none)
        1
        1   IMPLICIT INPUTS:
        1
        1     NONE
        1
        1   IMPLICIT OUTPUTS:
        1
        1     RABSB_RAC            record access field
        1
        1   ROUTINE VALUE:
        1
        1     NONE
        1
        1   SIDE EFFECTS:
        1
        1     Finds next record in file on this logical unit.
        1     SIGNALS any RMS errors
        1
        1   BEGIN
        1
        1   EXTERNAL REGISTER
        1     CCB : REF BLOCK [, BYTE];
        1
        1   LOCAL
        1     RMS_STATUS;
        1
        1   !+
        1     Set the record access field in the RAB to sequential. Perform the FIND.
        1     If RMS returns a failure status, signal the error.
        1
        1   CCB [RABSB_RAC] = RABSC_RFA;
        1
        1   !+
        1     Set bits in RAB ROP without clearing ULK.
        1
        1   CCB [RABSL_ROP] = .CCB [RABSL_ROP] OR .LOCK_FLAGS;
        1
        1   !+
        1     perform the FIND.
```

```

:: 3149    4415 2
:: 3150    4416 2 RMS_STATUS = $FIND (RAB = .CCB);
:: 3151    4417 2
:: 3152    4418 2
:: 3153    4419 2
:: 3154    4420 2
:: 3155    4421 2
:: 3156    4422 2 CCB [RABSL_ROP] = .CCB [RABSL_ROP] XOR .LOCK_FLAGS;
:: 3157    4423 2
:: 3158    4424 2
:: 3159    4425 2
:: 3160    4426 2
:: 3161    4427 2 IF NOT .RMS_STATUS
:: 3162    4428 2 THEN
:: 3163    4429 2     BASS$STOP_IO (BASSK_IOERR_REC);
:: 3164    4430 2
:: 3165    4431 2 RETURN;
:: 3166    4432 1 END;           ! End of BASS$REC_FRFA

```

		52 DD 00000 BASS\$REC_FRFA::		
		PUSHL R2		: 4358
1E	52 AB	MOVL R0, R2		: 4404
04	AB	MOVBL #2, 30(CCB)		: 4410
		BISL2 LOCK_FLAGS, 4(CCB)		: 4416
00000000G	00	PUSHL CCB		: 4422
04	AB	CALLS #1, SYSSFIND		: 4427
		XORL2 LOCK_FLAGS, 4(CCB)		: 4429
0A		BLBS RMS_STATUS, 1S		: 4432
00000000G	00	MNEGL #1, -(SP)		
7E		CALLS #1, BASS\$STOP_IO		
		POPR #^M<R2>		
		RSB		

; Routine Size: 42 bytes. Routine Base: _BASS\$CODE + 0B1A

; 3167 4433 1
; 3168 4434 1

```
3170 4435 1 GLOBAL ROUTINE BASS$REC_FRE ( ! FIND (relative) a record
3171 4436 1   LOCK_FLAGS
3172 4437 1   ) : JSB_REC0 NOVALUE =
3173 4438 1
3174 4439 1 ++
3175 4440 1   FUNCTIONAL DESCRIPTION:
3176 4441 1
3177 4442 1   Find next record. If successful then return; otherwise, signal a fatal
3178 4443 1   error.
3179 4444 1
3180 4445 1   FORMAL PARAMETERS:
3181 4446 1
3182 4447 1   LOCK_FLAGS.rlu.v bits to set in the RAB ROP to control manual
3183 4448 1   record locking (0 if none)
3184 4449 1
3185 4450 1   IMPLICIT INPUTS:
3186 4451 1
3187 4452 1   NONE
3188 4453 1
3189 4454 1   IMPLICIT OUTPUTS:
3190 4455 1
3191 4456 1   RABSB_RAC record access field
3192 4457 1
3193 4458 1   ROUTINE VALUE:
3194 4459 1
3195 4460 1   NONE
3196 4461 1
3197 4462 1   SIDE EFFECTS:
3198 4463 1
3199 4464 1   SIGNALS any RMS errors
3200 4465 1
3201 4466 1
3202 4467 2 BEGIN
3203 4468 2
3204 4469 2
3205 4470 2   EXTERNAL REGISTER
3206 4471 2   [CB : REF BLOCK [, BYTE];
3207 4472 2
3208 4473 2
3209 4474 2
3210 4475 2
3211 4476 2
3212 4477 2   LOCAL
3213 4478 2
3214 4479 2
3215 4480 2
3216 4481 2
3217 4482 2
3218 4483 2
3219 4484 2
3220 4485 2
3221 4486 2
3222 4487 2
3223 4488 2
3224 4489 2
3225 4490 2
3226 4491 2
      + Set the record access field in the RAB to sequential. Perform the FIND.
      | If RMS returns a failure status, signal the error.
      |
      CCB [RABSB_RAC] = RABSC_KEY;
      +
      | Set bits in the RAB ROP without clearing ULK.
      |
      CCB [RABSL_ROP] = .CCB [RABSL_ROP] OR .LOCK_FLAGS;
      +
      | perform the FIND.
```

```

: 3227    4492 2 RMS_STATUS = $FIND (RAB = .CCB);
: 3228    4493 2
: 3229    4494 2
: 3230    4495 2 !+ Turn off bits in the RAB ROP so that subsequent operations do not
: 3231    4496 2 inherit them.
: 3232    4497 2 !-
: 3233    4498 2
: 3234    4499 2 CCB [RAB$L_ROP] = .CCB [RAB$L_ROP] XOR .LOCK_FLAGS;
: 3235    4500 2
: 3236    4501 2
: 3237    4502 2 !+ Signal if the FIND failed.
: 3238    4503 2 !-
: 3239    4504 2 IF NOT .RMS_STATUS
: 3240    4505 2 THEN
: 3241    4506 2     BASS$STOP_IO (BASS$IOERR_REC);
: 3242    4507 2
: 3243    4508 2 RETURN;
: 3244    4509 1 END;

```

! End of BASS\$REC_FRE

1E AB	01 90 00000 BASS\$REC_FRE::		
04 AB	04 AE C8 00004	MOV B #1, 30(CC8)	4480
00000000G 00	5B DD 00009	BISL2 LOCK_FLAGS, 4(CC8)	4486
04 AB	01 FB 0000B	PUSHL CCB	4492
0A	AE CC 00012	CALLS #1, SYSSFIND	4499
00000000G 00	50 E8 00017	XORL2 LOCK_FLAGS, 4(CC8)	4504
7E	01 CE 0001A	BLBS RMS_STATUS, 1\$	4506
05 00024 1\$:	01 FB 0001D	MNEGL #1, -(SP)	
	05 00024 1\$:	CALLS #1, BASS\$STOP_IO	
		RSB	4509

: Routine Size: 37 bytes, Routine Base: _BASSCODE + 0B44

: 3245 4510 1

: 3247 4511 1 GLOBAL ROUTINE BASSREC_FIN (! FIND (indexed) a record
: 3248 4512 1 KEY_NO, REL_OP, KEY, LOCK_FLAGS) : JSB_REC_IND1 NOVALUE =
: 3249 4513 1
: 3250 4514 1
: 3251 4515 1
: 3252 4516 1
: 3253 4517 1
: 3254 4518 1
: 3255 4519 1
: 3256 4520 1
: 3257 4521 1
: 3258 4522 1
: 3259 4523 1
: 3260 4524 1
: 3261 4525 1
: 3262 4526 1
: 3263 4527 1
: 3264 4528 1
: 3265 4529 1
: 3266 4530 1
: 3267 4531 1
: 3268 4532 1
: 3269 4533 1
: 3270 4534 1
: 3271 4535 1
: 3272 4536 1
: 3273 4537 1
: 3274 4538 1
: 3275 4539 1
: 3276 4540 1
: 3277 4541 1
: 3278 4542 1
: 3279 4543 1
: 3280 4544 1
: 3281 4545 1
: 3282 4546 1
: 3283 4547 1
: 3284 4548 1
: 3285 4549 1
: 3286 4550 1
: 3287 4551 1
: 3288 4552 1
: 3289 4553 2
: 3290 4554 2
: 3291 4555 2
: 3292 4556 2
: 3293 4557 2
: 3294 4558 2
: 3295 4559 2
: 3296 4560 2
: 3297 4561 2
: 3298 4562 2
: 3299 4563 2
: 3300 4564 2
: 3301 4565 2
: 3302 4566 2
: 3303 4567 2

GLOBAL ROUTINE BASSREC_FIN (! FIND (indexed) a record
KEY_NO, REL_OP, KEY, LOCK_FLAGS) : JSB_REC_IND1 NOVALUE =
++
FUNCTIONAL DESCRIPTION:
Find indicated record. If successful then return; otherwise, signal a fatal error.
FORMAL PARAMETERS:
KEY_NO.rl.v key of reference
REL_OP.rl.v relational operator for key
KEY.rl.dx key to search for
LOCK_FLAGS.rlu.v bits to set in the RAB ROP to control manual record locking (0 if none)
IMPLICIT INPUTS:
NONE
IMPLICIT OUTPUTS:
RABSL_KBF pointer to the desired key value
RABSB_KSZ size of desired key value
RABSM_KGE relational in RABSL_ROP indicating greater than or equal
RABSM_KGT relational in RABSL_ROP indicating greater than
RABSB_KRF indicates key of reference
RABSB_RAC record access field
ROUTINE VALUE:
NONE
SIDE EFFECTS:
See RMS Reference manual for discussion on whether match will be exact, generic, approximate, or generic-approximate.
--
SIGNALS any RMS errors
--
BEGIN
EXTERNAL REGISTER
CCB : REF BLOCK [, BYTE];
MAP
KEY : REF BLOCK [8, BYTE];
LITERAL
K_EQUAL = 0, ! search for key equal
K_GREATER_EQUAL = 1, ! search for key GEQ
K_GREATER_THAN = 2; ! search for key GTR
LOCAL
RMS_STATUS;

```
: 3304      4568 2
: 3305      4569 2
: 3306      4570 2
: 3307      4571 2
: 3308      4572 2
: 3309      4573 2
: 3310      4574 2
: 3311      4575 2
: 3312      4576 2
: 3313      4577 2
: 3314      4578 2
: 3315      4579 2
: 3316      4580 2
: 3317      4581 2
: 3318      4582 2
: 3319      4583 2
: 3320      4584 2
: 3321      4585 2
: 3322      4586 2
: 3323      4587 2
: 3324      4588 2
: 3325      4589 2
: 3326      4590 2
: 3327      4591 2
: 3328      4592 2
: 3329      4593 2
: 3330      4594 2
: 3331      4595 2
: 3332      4596 2
: 3333      4597 2
: 3334      4598 2
: 3335      4599 2
: 3336      4600 2
: 3337      4601 2
: 3338      4602 2
: 3339      4603 2
: 3340      4604 2
: 3341      4605 2
: 3342      4606 2
: 3343      4607 2
: 3344      4608 2
: 3345      4609 2
: 3346      4610 2
: 3347      4611 2
: 3348      4612 2
: 3349      4613 2
: 3350      4614 2
: 3351      4615 2
: 3352      4616 2
: 3353      4617 2
: 3354      4618 2
: 3355      4619 2
: 3356      4620 2
: 3357      4621 2
: 3358      4622 2
: 3359      4623 2
: 3360      4624 2

        |+ Set the key buffer field, the key size field, the key of reference,
        |+ and the relational bits in the ROP.
        |+ Set the record access field in the RAB to key. Perform the FIND.
        |- If RMS returns a failure status, signal the error.

CCB[RABSB_RAC] = RABSC_KEY;
CCB[RABSL_KBF] = .KEY [DSC$A_POINTER];
CCB[RABSB_KRF] = .KEY_NO;
CCB[RABSB_KSZ] = {IF .KEY [DSC$B_DTYPE] NEQ DSC$K_DTYPE_P
                  THEN
                    .KEY [DSC$W_LENGTH]
                  ELSE
                    (.KEY [DSC$W_LENGTH]/2 + 1)}:

CASE .REL_OP FROM K_EQUAL TO K_GREATER_THAN OF
  SET
    [K_EQUAL] :
      CCB[RABSV_KGE] = CCB[RABSV_KGT] = 0;
    [K_GREATER_EQUAL] :
      BEGIN
        CCB[RABSV_KGE] = 1;
        CCB[RABSV_KGT] = 0;
      END;
    [K_GREATER_THAN] :
      BEGIN
        CCB[RABSV_KGT] = 1;
        CCB[RABSV_KGE] = 0;
      END;
  TES;

        |+ Set bits in the RAB ROP without clearing ULK.
        |-  

CCB[RABSL_ROP] = .CCB[RABSL_ROP] OR .LOCK_FLAGS;
        |+ perform the FIND.

RMS_STATUS = SFIND (RAB = .CCB);

        |+ Turn off bits in the RAB ROP so that subsequent operations do not
        |+ inherit them.
        |-  

CCB[RABSL_ROP] = .CCB[RABSL_ROP] XOR .LOCK_FLAGS;
        |+ Signal if the FIND failed.
```

```

: 3361    4625  2      :-  

: 3362    4626  2      IF NOT .RMS_STATUS  

: 3363    4627  2      THEN  

: 3364    4628  2      BASS$STOP_IO (BASSK_IOERR_REC);  

: 3365    4629  2      RETURN;  

: 3366    4630  2      END;
: 3367    4631  1

```

! End of BASS\$REC_FIN

52 DD 00000 BASS\$REC FIN::										
1E	AB		04	01	90 00002		PUSHL	R2		4511
30	AB			A2	D0 00006		MOV8	#1, 30(CCB)		4576
35	AB		02	50	90 00008		MOVL	4(KEY), 48(CCB)		4577
	15			A2	91 0000F		MOV8	KEY NO. 53(CCB)		4578
				05	13 00013		CMPB	2(KEY), #21		4579
				62	3C 00015		BEQL	1\$		4581
				08	11 00018		MOVZWL	(KEY), R2		4583
				62	3C 0001A	1\$:	BRB	2\$		
				52	C6 0001D		MOVZWL	(KEY), R2		
				52	D6 00020		DIVL2	#2, R2		
				52	90 00022	2\$:	INCL	R2		
	34	AB		52	AB 00026	2\$:	MOV8	R2, 52(CCB)		4579
	52		04	51	CF 0002A	3\$:	MOVAB	4(CCB), R2		4589
02	00	0000		0006	0002E	3\$:	CASEL	REL OP, #0, #2		4585
							.WORD	48-3\$, -		
								58-3\$, -		
								68-3\$		
				02	A2	40	BICB2	#64, 2(R2)		4589
						10	BRB	7\$		
				02	A2	20	BISB2	#32, 2(R2)		4593
				02	A2	8F	BICB2	#64, 2(R2)		4594
				02	A2	09	BRB	8\$		4585
				02	A2	8F	BISB2	#64, 2(R2)		4599
				02	A2	20	BICB2	#32, 2(R2)		4600
					62	53	BISL2	LOCK_FLAGS, (R2)		4608
						5B	PUSHL	CCB		4614
				00000000G	00	01	CALLS	#1, SYSSFIND		
					62	FB 00054	XORL2	LOCK_FLAGS, (R2)		4621
					0A	53 CC 0005B	BLBS	RMS_STATUS, 9\$		4626
				00000000G	00	50 EB 0005E	MNEGL	#1, -(SP)		4628
					7E	01 CE 00061	CALLS	#1, BASS\$STOP_IO		
						01 FB 00064	POPR	#^M<R2>		
						04 BA 0006B	RSB			4631

: Routine Size: 110 bytes. Routine Base: _BASSCODE + 0B69

: 3368 4632 1

```

: 3370      4633 1 GLOBAL ROUTINE BASS$REC_DSE           ! DELETE (sequential) a record
: 3371      4634 1 : JSB_REC NOVALUE =
: 3372      4635 1
: 3373      4636 1
: 3374      4637 1
: 3375      4638 1
: 3376      4639 1 FUNCTIONAL DESCRIPTION:
: 3377      4640 1 Delete current record. If successful then return; otherwise, signal a fatal
: 3378      4641 1 error.
: 3379      4642 1 FORMAL PARAMETERS:
: 3380      4643 1
: 3381      4644 1     NONE
: 3382      4645 1
: 3383      4646 1 IMPLICIT INPUTS:
: 3384      4647 1     NONE
: 3385      4648 1
: 3386      4649 1 IMPLICIT OUTPUTS:
: 3387      4650 1     RABSB_RAC          record access field
: 3388      4651 1
: 3389      4652 1
: 3390      4653 1
: 3391      4654 1 ROUTINE VALUE:
: 3392      4655 1     NONE
: 3393      4656 1
: 3394      4657 1
: 3395      4658 1 SIDE EFFECTS:
: 3396      4659 1
: 3397      4660 1     SIGNALS any RMS errors
: 3398      4661 1
: 3399      4662 1
: 3400      4663 2
: 3401      4664 2
: 3402      4665 2
: 3403      4666 2 EXTERNAL REGISTER
: 3404      4667 2     CCB : REF BLOCK [. BYTE];
: 3405      4668 2
: 3406      4669 2
: 3407      4670 2
: 3408      4671 2
: 3409      4672 2
: 3410      4673 2
: 3411      4674 2
: 3412      4675 2
: 3413      4676 2
: 3414      4677 2
: 3415      4678 2
: 3416      4679 2
: 3417      4680 2
: 3418      4681 2
: 3419      4682 2
: 3420      4683 2
: 3421      4684 2
: 3422      4685 2
: 3423      4686 2
: 3424      4687 2
: 3425      4688 2
: 3426      4689 2
: 3427      4690 2
: 3428      4691 2
: 3429      4692 2
: 3430      4693 2
: 3431      4694 2
: 3432      4695 2
: 3433      4696 2
: 3434      4697 2
: 3435      4698 2
: 3436      4699 2
: 3437      4700 2
: 3438      4701 2
: 3439      4702 2
: 3440      4703 2
: 3441      4704 2
: 3442      4705 2
: 3443      4706 2
: 3444      4707 2
: 3445      4708 2
: 3446      4709 2
: 3447      4710 2
: 3448      4711 2
: 3449      4712 2
: 3450      4713 2
: 3451      4714 2
: 3452      4715 2
: 3453      4716 2
: 3454      4717 2
: 3455      4718 2
: 3456      4719 2
: 3457      4720 2
: 3458      4721 2
: 3459      4722 2
: 3460      4723 2
: 3461      4724 2
: 3462      4725 2
: 3463      4726 2
: 3464      4727 2
: 3465      4728 2
: 3466      4729 2
: 3467      4730 2
: 3468      4731 2
: 3469      4732 2
: 3470      4733 2
: 3471      4734 2
: 3472      4735 2
: 3473      4736 2
: 3474      4737 2
: 3475      4738 2
: 3476      4739 2
: 3477      4740 2
: 3478      4741 2
: 3479      4742 2
: 3480      4743 2
: 3481      4744 2
: 3482      4745 2
: 3483      4746 2
: 3484      4747 2
: 3485      4748 2
: 3486      4749 2
: 3487      4750 2
: 3488      4751 2
: 3489      4752 2
: 3490      4753 2
: 3491      4754 2
: 3492      4755 2
: 3493      4756 2
: 3494      4757 2
: 3495      4758 2
: 3496      4759 2
: 3497      4760 2
: 3498      4761 2
: 3499      4762 2
: 3500      4763 2
: 3501      4764 2
: 3502      4765 2
: 3503      4766 2
: 3504      4767 2
: 3505      4768 2
: 3506      4769 2
: 3507      4770 2
: 3508      4771 2
: 3509      4772 2
: 3510      4773 2
: 3511      4774 2
: 3512      4775 2
: 3513      4776 2
: 3514      4777 2
: 3515      4778 2
: 3516      4779 2
: 3517      4780 2
: 3518      4781 2
: 3519      4782 2
: 3520      4783 2
: 3521      4784 2
: 3522      4785 2
: 3523      4786 2
: 3524      4787 2
: 3525      4788 2
: 3526      4789 2
: 3527      4790 2
: 3528      4791 2
: 3529      4792 2
: 3530      4793 2
: 3531      4794 2
: 3532      4795 2
: 3533      4796 2
: 3534      4797 2
: 3535      4798 2
: 3536      4799 2
: 3537      4800 2
: 3538      4801 2
: 3539      4802 2
: 3540      4803 2
: 3541      4804 2
: 3542      4805 2
: 3543      4806 2
: 3544      4807 2
: 3545      4808 2
: 3546      4809 2
: 3547      4810 2
: 3548      4811 2
: 3549      4812 2
: 3550      4813 2
: 3551      4814 2
: 3552      4815 2
: 3553      4816 2
: 3554      4817 2
: 3555      4818 2
: 3556      4819 2
: 3557      4820 2
: 3558      4821 2
: 3559      4822 2
: 3560      4823 2
: 3561      4824 2
: 3562      4825 2
: 3563      4826 2
: 3564      4827 2
: 3565      4828 2
: 3566      4829 2
: 3567      4830 2
: 3568      4831 2
: 3569      4832 2
: 3570      4833 2
: 3571      4834 2
: 3572      4835 2
: 3573      4836 2
: 3574      4837 2
: 3575      4838 2
: 3576      4839 2
: 3577      4840 2
: 3578      4841 2
: 3579      4842 2
: 3580      4843 2
: 3581      4844 2
: 3582      4845 2
: 3583      4846 2
: 3584      4847 2
: 3585      4848 2
: 3586      4849 2
: 3587      4850 2
: 3588      4851 2
: 3589      4852 2
: 3590      4853 2
: 3591      4854 2
: 3592      4855 2
: 3593      4856 2
: 3594      4857 2
: 3595      4858 2
: 3596      4859 2
: 3597      4860 2
: 3598      4861 2
: 3599      4862 2
: 3600      4863 2
: 3601      4864 2
: 3602      4865 2
: 3603      4866 2
: 3604      4867 2
: 3605      4868 2
: 3606      4869 2
: 3607      4870 2
: 3608      4871 2
: 3609      4872 2
: 3610      4873 2
: 3611      4874 2
: 3612      4875 2
: 3613      4876 2
: 3614      4877 2
: 3615      4878 2
: 3616      4879 2
: 3617      4880 2
: 3618      4881 2
: 3619      4882 2
: 3620      4883 2
: 3621      4884 2
: 3622      4885 2
: 3623      4886 2
: 3624      4887 2
: 3625      4888 2
: 3626      4889 2
: 3627      4890 2
: 3628      4891 2
: 3629      4892 2
: 3630      4893 2
: 3631      4894 2
: 3632      4895 2
: 3633      4896 2
: 3634      4897 2
: 3635      4898 2
: 3636      4899 2
: 3637      4900 2
: 3638      4901 2
: 3639      4902 2
: 3640      4903 2
: 3641      4904 2
: 3642      4905 2
: 3643      4906 2
: 3644      4907 2
: 3645      4908 2
: 3646      4909 2
: 3647      4910 2
: 3648      4911 2
: 3649      4912 2
: 3650      4913 2
: 3651      4914 2
: 3652      4915 2
: 3653      4916 2
: 3654      4917 2
: 3655      4918 2
: 3656      4919 2
: 3657      4920 2
: 3658      4921 2
: 3659      4922 2
: 3660      4923 2
: 3661      4924 2
: 3662      4925 2
: 3663      4926 2
: 3664      4927 2
: 3665      4928 2
: 3666      4929 2
: 3667      4930 2
: 3668      4931 2
: 3669      4932 2
: 3670      4933 2
: 3671      4934 2
: 3672      4935 2
: 3673      4936 2
: 3674      4937 2
: 3675      4938 2
: 3676      4939 2
: 3677      4940 2
: 3678      4941 2
: 3679      4942 2
: 3680      4943 2
: 3681      4944 2
: 3682      4945 2
: 3683      4946 2
: 3684      4947 2
: 3685      4948 2
: 3686      4949 2
: 3687      4950 2
: 3688      4951 2
: 3689      4952 2
: 3690      4953 2
: 3691      4954 2
: 3692      4955 2
: 3693      4956 2
: 3694      4957 2
: 3695      4958 2
: 3696      4959 2
: 3697      4960 2
: 3698      4961 2
: 3699      4962 2
: 3700      4963 2
: 3701      4964 2
: 3702      4965 2
: 3703      4966 2
: 3704      4967 2
: 3705      4968 2
: 3706      4969 2
: 3707      4970 2
: 3708      4971 2
: 3709      4972 2
: 3710      4973 2
: 3711      4974 2
: 3712      4975 2
: 3713      4976 2
: 3714      4977 2
: 3715      4978 2
: 3716      4979 2
: 3717      4980 2
: 3718      4981 2
: 3719      4982 2
: 3720      4983 2
: 3721      4984 2
: 3722      4985 2
: 3723      4986 2
: 3724      4987 2
: 3725      4988 2
: 3726      4989 2
: 3727      4990 2
: 3728      4991 2
: 3729      4992 2
: 3730      4993 2
: 3731      4994 2
: 3732      4995 2
: 3733      4996 2
: 3734      4997 2
: 3735      4998 2
: 3736      4999 2
: 3737      5000 2
: 3738      5001 2
: 3739      5002 2
: 3740      5003 2
: 3741      5004 2
: 3742      5005 2
: 3743      5006 2
: 3744      5007 2
: 3745      5008 2
: 3746      5009 2
: 3747      5010 2
: 3748      5011 2
: 3749      5012 2
: 3750      5013 2
: 3751      5014 2
: 3752      5015 2
: 3753      5016 2
: 3754      5017 2
: 3755      5018 2
: 3756      5019 2
: 3757      5020 2
: 3758      5021 2
: 3759      5022 2
: 3760      5023 2
: 3761      5024 2
: 3762      5025 2
: 3763      5026 2
: 3764      5027 2
: 3765      5028 2
: 3766      5029 2
: 3767      5030 2
: 3768      5031 2
: 3769      5032 2
: 3770      5033 2
: 3771      5034 2
: 3772      5035 2
: 3773      5036 2
: 3774      5037 2
: 3775      5038 2
: 3776      5039 2
: 3777      5040 2
: 3778      5041 2
: 3779      5042 2
: 3780      5043 2
: 3781      5044 2
: 3782      5045 2
: 3783      5046 2
: 3784      5047 2
: 3785      5048 2
: 3786      5049 2
: 3787      5050 2
: 3788      5051 2
: 3789      5052 2
: 3790      5053 2
: 3791      5054 2
: 3792      5055 2
: 3793      5056 2
: 3794      5057 2
: 3795      5058 2
: 3796      5059 2
: 3797      5060 2
: 3798      5061 2
: 3799      5062 2
: 3800      5063 2
: 3801      5064 2
: 3802      5065 2
: 3803      5066 2
: 3804      5067 2
: 3805      5068 2
: 3806      5069 2
: 3807      5070 2
: 3808      5071 2
: 3809      5072 2
: 3810      5073 2
: 3811      5074 2
: 3812      5075 2
: 3813      5076 2
: 3814      5077 2
: 3815      5078 2
: 3816      5079 2
: 3817      5080 2
: 3818      5081 2
: 3819      5082 2
: 3820      5083 2
: 3821      5084 2
: 3822      5085 2
: 3823      5086 2
: 3824      5087 2
: 3825      5088 2
: 3826      5089 2
: 3827      5090 2
: 3828      5091 2
: 3829      5092 2
: 3830      5093 2
: 3831      5094 2
: 3832      5095 2
: 3833      5096 2
: 3834      5097 2
: 3835      5098 2
: 3836      5099 2
: 3837      5100 2
: 3838      5101 2
: 3839      5102 2
: 3840      5103 2
: 3841      5104 2
: 3842      5105 2
: 3843      5106 2
: 3844      5107 2
: 3845      5108 2
: 3846      5109 2
: 3847      5110 2
: 3848      5111 2
: 3849      5112 2
: 3850      5113 2
: 3851      5114 2
: 3852      5115 2
: 3853      5116 2
: 3854      5117 2
: 3855      5118 2
: 3856      5119 2
: 3857      5120 2
: 3858      5121 2
: 3859      5122 2
: 3860      5123 2
: 3861      5124 2
: 3862      5125 2
: 3863      5126 2
: 3864      5127 2
: 3865      5128 
```

BASS\$REC_PROC
1-095

G 11
16-Sep-1984 01:01:12 VAX-11 Bliss-32 V4.0-742
14-Sep-1984 11:56:35 [BASRTL.SRC]BASREC[PRO.B32;1]

Page 92
(34)

00000000G 00 01 FB 00012 CALLS #1, BASS\$STOP_10
05 00019 1\$: RSB

; 4678

; Routine Size: 26 bytes, Routine Base: _BASSCODE + 0BD7

; 3416 4679 1

```

3418 4680 1 GLOBAL ROUTINE BASS$REC_UNL           ! UNLOCK a record
3419 4681 1 : JSB_REC0 NOVALUE ≡
3420 4682 1
3421 4683 1 ++
3422 4684 1   FUNCTIONAL DESCRIPTION:
3423 4685 1
3424 4686 1     Unlock the current record. If successful or no records locked,
3425 4687 1     then return; otherwise, signal a fatal error.
3426 4688 1
3427 4689 1   FORMAL PARAMETERS:
3428 4690 1
3429 4691 1     NONE
3430 4692 1
3431 4693 1   IMPLICIT INPUTS:
3432 4694 1
3433 4695 1     NONE
3434 4696 1
3435 4697 1   IMPLICIT OUTPUTS:
3436 4698 1
3437 4699 1     RABSB_RAC          record access field
3438 4700 1
3439 4701 1   ROUTINE VALUE:
3440 4702 1
3441 4703 1     NONE
3442 4704 1
3443 4705 1   SIDE EFFECTS:
3444 4706 1
3445 4707 1     SIGNALS any RMS errors
3446 4708 1
3447 4709 1
3448 4710 2   BEGIN
3449 4711 2
3450 4712 2   EXTERNAL REGISTER
3451 4713 2     CCB : REF BLOCK [. BYTE];
3452 4714 2
3453 4715 2
3454 4716 2   |+
3455 4717 2     | Set the record access field in the RAB to sequential. Perform the UNLOCK.
3456 4718 2     | If RMS returns a failure status, signal the error.
3457 4719 2   |-
3458 4720 2
3459 4721 2
3460 4722 2   CCB [RABSB_RAC] = RABSC_SEQ;
3461 4723 2
3462 4724 2
3463 4725 2
3464 4726 2
3465 4727 2
3466 4728 2
3467 4729 2
3468 4730 2
3469 4731 2
3470 4732 2
3471 4733 2
3472 4734 2
3473 4735 2
3474 4736 2
3475 4737 2
3476 4738 2
3477 4739 2
3478 4740 2
3479 4741 2
3480 4742 2
3481 4743 2
3482 4744 2
3483 4745 2
3484 4746 2
3485 4747 2
3486 4748 2
3487 4749 2
3488 4750 2
3489 4751 2
3490 4752 2
3491 4753 2
3492 4754 2
3493 4755 2
3494 4756 2
3495 4757 2
3496 4758 2
3497 4759 2
3498 4760 2
3499 4761 2
3500 4762 2
3501 4763 2
3502 4764 2
3503 4765 2
3504 4766 2
3505 4767 2
3506 4768 2
3507 4769 2
3508 4770 2
3509 4771 2
3510 4772 2
3511 4773 2
3512 4774 2
3513 4775 2
3514 4776 2
3515 4777 2
3516 4778 2
3517 4779 2
3518 4780 2
3519 4781 2
3520 4782 2
3521 4783 2
3522 4784 2
3523 4785 2
3524 4786 2
3525 4787 2
3526 4788 2
3527 4789 2
3528 4790 2
3529 4791 2
3530 4792 2
3531 4793 2
3532 4794 2
3533 4795 2
3534 4796 2
3535 4797 2
3536 4798 2
3537 4799 2
3538 4800 2
3539 4801 2
3540 4802 2
3541 4803 2
3542 4804 2
3543 4805 2
3544 4806 2
3545 4807 2
3546 4808 2
3547 4809 2
3548 4810 2
3549 4811 2
3550 4812 2
3551 4813 2
3552 4814 2
3553 4815 2
3554 4816 2
3555 4817 2
3556 4818 2
3557 4819 2
3558 4820 2
3559 4821 2
3560 4822 2
3561 4823 2
3562 4824 2
3563 4825 2
3564 4826 2
3565 4827 2
3566 4828 2
3567 4829 2
3568 4830 2
3569 4831 2
3570 4832 2
3571 4833 2
3572 4834 2
3573 4835 2
3574 4836 2
3575 4837 2
3576 4838 2
3577 4839 2
3578 4840 2
3579 4841 2
3580 4842 2
3581 4843 2
3582 4844 2
3583 4845 2
3584 4846 2
3585 4847 2
3586 4848 2
3587 4849 2
3588 4850 2
3589 4851 2
3590 4852 2
3591 4853 2
3592 4854 2
3593 4855 2
3594 4856 2
3595 4857 2
3596 4858 2
3597 4859 2
3598 4860 2
3599 4861 2
3600 4862 2
3601 4863 2
3602 4864 2
3603 4865 2
3604 4866 2
3605 4867 2
3606 4868 2
3607 4869 2
3608 4870 2
3609 4871 2
3610 4872 2
3611 4873 2
3612 4874 2
3613 4875 2
3614 4876 2
3615 4877 2
3616 4878 2
3617 4879 2
3618 4880 2
3619 4881 2
3620 4882 2
3621 4883 2
3622 4884 2
3623 4885 2
3624 4886 2
3625 4887 2
3626 4888 2
3627 4889 2
3628 4890 2
3629 4891 2
3630 4892 2
3631 4893 2
3632 4894 2
3633 4895 2
3634 4896 2
3635 4897 2
3636 4898 2
3637 4899 2
3638 4900 2
3639 4901 2
3640 4902 2
3641 4903 2
3642 4904 2
3643 4905 2
3644 4906 2
3645 4907 2
3646 4908 2
3647 4909 2
3648 4910 2
3649 4911 2
3650 4912 2
3651 4913 2
3652 4914 2
3653 4915 2
3654 4916 2
3655 4917 2
3656 4918 2
3657 4919 2
3658 4920 2
3659 4921 2
3660 4922 2
3661 4923 2
3662 4924 2
3663 4925 2
3664 4926 2
3665 4927 2
3666 4928 2
3667 4929 2
3668 4930 2
3669 4931 2
3670 4932 2
3671 4933 2
3672 4934 2
3673 4935 2
3674 4936 2
3675 4937 2
3676 4938 2
3677 4939 2
3678 4940 2
3679 4941 2
3680 4942 2
3681 4943 2
3682 4944 2
3683 4945 2
3684 4946 2
3685 4947 2
3686 4948 2
3687 4949 2
3688 4950 2
3689 4951 2
3690 4952 2
3691 4953 2
3692 4954 2
3693 4955 2
3694 4956 2
3695 4957 2
3696 4958 2
3697 4959 2
3698 4960 2
3699 4961 2
3700 4962 2
3701 4963 2
3702 4964 2
3703 4965 2
3704 4966 2
3705 4967 2
3706 4968 2
3707 4969 2
3708 4970 2
3709 4971 2
3710 4972 2
3711 4973 2
3712 4974 2
3713 4975 2
3714 4976 2
3715 4977 2
3716 4978 2
3717 4979 2
3718 4980 2
3719 4981 2
3720 4982 2
3721 4983 2
3722 4984 2
3723 4985 2
3724 4986 2
3725 4987 2
3726 4988 2
3727 4989 2
3728 4990 2
3729 4991 2
3730 4992 2
3731 4993 2
3732 4994 2
3733 4995 2
3734 4996 2
3735 4997 2
3736 4998 2
3737 4999 2
3738 5000 2
3739 5001 2
3740 5002 2
3741 5003 2
3742 5004 2
3743 5005 2
3744 5006 2
3745 5007 2
3746 5008 2
3747 5009 2
3748 5010 2
3749 5011 2
3750 5012 2
3751 5013 2
3752 5014 2
3753 5015 2
3754 5016 2
3755 5017 2
3756 5018 2
3757 5019 2
3758 5020 2
3759 5021 2
3760 5022 2
3761 5023 2
3762 5024 2
3763 5025 2
3764 5026 2
3765 5027 2
3766 5028 2
3767 5029 2
3768 5030 2
3769 5031 2
3770 5032 2
3771 5033 2
3772 5034 2
3773 5035 2
3774 5036 2
3775 5037 2
3776 5038 2
3777 5039 2
3778 5040 2
3779 5041 2
3780 5042 2
3781 5043 2
3782 5044 2
3783 5045 2
3784 5046 2
3785 5047 2
3786 5048 2
3787 5049 2
3788 5050 2
3789 5051 2
3790 5052 2
3791 5053 2
3792 5054 2
3793 5055 2
3794 5056 2
3795 5057 2
3796 5058 2
3797 5059 2
3798 5060 2
3799 5061 2
3800 5062 2
3801 5063 2
3802 5064 2
3803 5065 2
3804 5066 2
3805 5067 2
3806 5068 2
3807 5069 2
3808 5070 2
3809 5071 2
3810 5072 2
3811 5073 2
3812 5074 2
3813 5075 2
3814 5076 2
3815 5077 2
3816 5078 2
3817 5079 2
3818 5080 2
3819 5081 2
3820 5082 2
3821 5083 2
3822 5084 2
3823 5085 2
3824 5086 2
3825 5087 2
3826 5088 2
3827 5089 2
3828 5090 2
3829 5091 2
3830 5092 2
3831 5093 2
3832 5094 2
3833 5095 2
3834 5096 2
3835 5097 2
3836 5098 2
3837 5099 2
3838 5100 2
3839 5101 2
3840 5102 2
3841 5103 2
3842 5104 2
3843 5105 2
3844 5106 2
3845 5107 2
3846 5108 2
3847 5109 2
3848 5110 2
3849 5111 2
3850 5112 2
3851 5113 2
3852 5114 2
3853 5115 2
3854 5116 2
3855 5117 2
3856 5118 2
3857 5119 2
3858 5120 2
3859 5121 2
3860 5122 2
3861 5123 2
3862 5124 2
3863 5125 2
3864 5126 2
3865 5127 2
3866 5128 2
3867 5129 2
3868 5130 2
3869 5131 2
3870 5132 2
3871 5133 2
3872 5134 2
3873 5135 2
3874 5136 2
3875 5137 2
3876 5138 2
3877 5139 2
3878 5140 2
3879 5141 2
3880 5142 2
3881 5143 2
3882 5144 2
3883 5145 2
3884 5146 2
3885 5147 2
3886 5148 2
3887 5149 2
3888 5150 2
3889 5151 2
3890 5152 2
3891 5153 2
3892 5154 2
3893 5155 2
3894 5156 2
3895 5157 2
3896 5158 2
3897 5159 2
3898 5160 2
3899 5161 2
3900 5162 2
3901 5163 2
3902 5164 2
3903 5165 2
3904 5166 2
3905 5167 2
3906 5168 2
3907 5169 2
3908 5170 2
3909 5171 2
3910 5172 2
3911 5173 2
3912 5174 2
3913 5175 2
3914 5176 2
3915 5177 2
3916 5178 2
3917 5179 2
3918 5180 2
3919 5181 2
3920 5182 2
3921 5183 2
3922 5184 2
3923 5185 2
3924 5186 2
3925 5187 2
3926 5188 2
3927 5189 2
3928 5190 2
3929 5191 2
3930 5192 2
3931 5193 2
3932 5194 2
3933 5195 2
3934 5196 2
3935 5197 2
3936 5198 2
3937 5199 2
3938 5200 2
3939 5201 2
3940 5202 2
3941 5203 2
3942 5204 2
3943 5205 2
3944 5206 2
3945 5207 2
3946 5208 2
3947 5209 2
3948 5210 2
3949 5211 2
3950 5212 2
3951 5213 2
3952 5214 2
3953 5215 2
3954 5216 2
3955 5217 2
3956 5218 2
3957 5219 2
3958 5220 2
3959 5221 2
3960 5222 2
3961 5223 2
3962 5224 2
3963 5225 2
3964 5226 2
3965 5227 2
3966 5228 2
3967 5229 2
3968 5230 2
3969 5231 2
3970 5232 2
3971 5233 2
3972 5234 2
3973 5235 2
3974 5236 2
3975 5237 2
3976 5238 2
3977 5239 2
3978 5240 2
3979 5241 2
3980 5242 2
3981 5243 2
3982 5244 2
3983 5245 2
3984 5246 2
3985 5247 2
3986 5248 2
3987 5249 2
3988 5250 2
3989 5251 2
3990 5252 2
3991 5253 2
3992 5254 2
3993 5255 2
3994 5256 2
3995 5257 2
3996 5258 2
3997 5259 2
3998 5260 2
3999 5261 2
4000 5262 2
4001 5263 2
4002 5264 2
4003 5265 2
4004 5266 2
4005 5267 2
4006 5268 2
4007 5269 2
4008 5270 2
4009 5271 2
4010 5272 2
4011 5273 2
4012 5274 2
4013 5275 2
4014 5276 2
4015 5277 2
4016 5278 2
4017 5279 2
4018 5280 2
4019 5281 2
4020 5282 2
4021 5283 2
4022 5284 2
4023 5285 2
4024 5286 2
4025 5287 2
4026 5288 2
4027 5289 2
4028 5290 2
4029 5291 2
4030 5292 2
4031 5293 2
4032 5294 2
4033 5295 2
4034 5296 2
4035 5297 2
4036 5298 2
4037 5299 2
4038 5300 2
4039 5301 2
4040 5302 2
4041 5303 2
4042 5304 2
4043 5305 2
4044 5306 2
4045 5307 2
4046 5308 2
4047 5309 2
4048 5310 2
4049 5311 2
4050 5312 2
4051 5313 2
4052 5314 2
4053 5315 2
4054 5316 2
4055 5317 2
4056 5318 2
4057 5319 2
4058 5320 2
4059 5321 2
4060 5322 2
4061 5323 2
4062 5324 2
4063 5325 2
4064 5326 2
4065 5327 2
4066 5328 2
4067 5329 2
4068 5330 2
4069 5331 2
4070 5332 2
4071 5333 2
4072 5334 2
4073 5335 2
4074 5336 2
4075 5337 2
4076 5338 2
4077 5339 2
4078 5340 2
4079 5341 2
4080 5342 2
4081 5343 2
4082 5344 2
4083 5345 2
4084 5346 2
4085 5347 2
4086 5348 2
4087 5349 2
4088 5350 2
4089 5351 2
4090 5352 2
4091 5353 2
4092 5354 2
4093 5355 2
4094 5356 2
4095 5357 2
4096 5358 2
4097 5359 2
4098 5360 2
4099 5361 2
4100 5362 2
4101 5363 2
4102 5364 2
4103 5365 2
4104 5366 2
4105 5367 2
4106 5368 2
4107 5369 2
4108 5370 2
4109 5371 2
4110 5372 2
4111 5373 2
4112 5374 2
4113 5375 2
4114 5376 2
4115 5377 2
4116 5378 2
4117 5379 2
4118 5380 2
4119 5381 2
4120 5382 2
4121 5383 2
4122 5384 2
4123 5385 2
4124 5386 2
4125 5387 2
4126 5388 2
4127 5389 2
4128 5390 2
4129 5391 2
4130 5392 2
4131 5393 2
4132 5394 2
4133 5395 2
4134 5396 2
4135 5397 2
4136 5398 2
4137 5399 2
4138 5400 2
4139 5401 2
4140 5402 2
4141 5403 2
4142 5404 2
4143 5405 2
4144 5406 2
4145 5407 2
4146 5408 2
4147 5409 2
4148 5410 2
4149 5411 2
4150 5412 2
4151 5413 2
4152 5414 2
4153 5415 2
4154 5416 2
4155 5417 2
4156 5418 2
4157 5419 2
4158 5420 2
4159 5421 2
4160 5422 2
4161 5423 2
4162 5424 2
4163 5425 2
4164 5426 2
4165 5427 2
4166 5428 2
4167 5429 2
4168 5430 2
4169 5431 2
4170 5432 2
4171 5433 2
4172 5434 2
4173 5435 2
4174 5436 2
4175 5437 2
4176 5438 2
4177 5439 2
4178 5440 2
4179 5441 2
4180 5442 2
4181 5443 2
4182 5444 2
4183 5445 2
4184 5446 2
4185 5447 2
4186 5448 2
4187 5449 2
4188 5450 2
4189 5451 2
4190 5452 2
4191 5453 2
4192 5454 2
4193 5455 2
4194 5456 2
4195 5457 2
4196 5458 2
4197 5459 2
4198 5460 2
4199 5461 2
4200 5462 2
4201 5463 2
4202 5464 2
4203 5465 2
4204 5466 2
4205 5467 2
4206 5468 2
4207 5469 2
4208 5470 2
4209 5471 2
4210 5472 2
4211 5473 2
4212 5474 2
4213 5475 2
4214 5476 2
4215 5477 2
4216 5478 2
4217 5479 2
4218 5480 2
4219 5481 2
4220 5482 2
4221 5483 2
4222 5484 2
4223 5485 2
4224 5486 2
4225 5487 2
4226 5488 2
4227 5489 2
4228 5490 2
4229 5491 2
4230 5492 2
4231 5493 2
4232 5494 2
4233 5495 2
4234 5496 2
4235 5497 2
4236 5498 2
4237 5499 2
4238 5500 2
4239 5501 2
4240 5502 2
4241 5503 2
4242 5504 2
4243 5505 2
4244 5506 2
4245 5507 2
4246 5508 2
4247 5509 2
4248 5510 2
4249 5511 2
4250 5512 2
4251 5513 2
4252 5514 2
4253 5515 2
4254 5516 2
4255 5517 2
4256 5518 2
4257 5519 2
4258 5520 2
4259 5521 2
4260 5522 2
4261 5523 2
4262 5524 2
4263 5525 2
4264 5526 2
4265 5527 2
4266 5528 2
4267 5529 2
4268 5530 2
4269 5531 2
4270 5532 2
4271 5533 2
4272 5534 2
4273 5535 2
4274 5536 2
4275 5537 2
4276 5538 2
4277 5539 2
4278 5540 2
4279 5541 2
4280 5542 2
4281 5543 2
4282 5544 2
4283 5545 2
4284 5546 2
4285 5547 2
4286 5548 2
4287 5549 2
4288 5550 2
4289 5551 2
4290 5552 2
4291 5553 2
4292 5554 2
4293 5555 2
4294 5556 2
4295 5557 2
4296 5558 2
4297 5559 2
4298 5560 2
4299 5561 2
4300 5562 2
4301 5563 2
4302 5564 2
4303 5565 2
4304 5566 2
4305 5567 2
4306 5568 2
4307 5569 2
4308 5570 2
4309 5571 2
4310 5572 2
4311 5573 2
4312 5574 2
4313 5575 2
4314 5576 2
4315 5577 2
4316 5578 2
4317 5579 2
4318 5580 2
4319 5581 2
4320 5582 2
4321 5583 2
4322 5584 2
4323 5585 2
4324 5586 2
4325 5587 2
4326 5588 2
4327 5589 2
4328 5590 2
4329 5591 2
4330 5592 2
4331 5593 2
4332 5594 2
4333 5595 2
4334 5596 2
4335 5597 2
4336 5598 2
4337 5599 2
4338 5600 2
4339 5601 2
4340 5602 2
4341 5603 2
4342 5604 2
4343 5605 2
4344 5606 2
4345 5607 2
4346 5608 2
4347 5609 2
4348 5610 2
4349 5611 2
4350 5612 2
4351 5613 2
4352 5614 2
4353 5615 2
4354 5616 2
4355 5617 2
4356 5618 2
4357 5619 2
4358 5620 2
4359 5621 2
4360 5622 
```

.EXTRN SYSSRELEASE

	1E	AB	94 00000 BASS\$REC UNL::		
		58	DD 00003	CLRB	30(CCB)
0000000G	00	01	FB 00005	PUSHL	CCB
	14	50	E8 0000C	CALLS	#1, SYSSRELEASE
000181A0	BF	08	A8 D1 0000F	BLBS	R0 1\$
		0A	13 00017	CMPL	8(CCB), #98720
0000000G	00	01	CE 00019	BEQL	1\$
	7E	01	FB 0001C	MNEGL	#1, -(SP)
		05	00023 1\$:	CALLS	#1, BASS\$STOP_10
				RSB	

; Routine Size: 36 bytes, Routine Base: _BASS\$CODE + 0BF1

; 3472 4734 1

3474 4735 1 GLOBAL ROUTINE BASS\$REC_FEE ! FREE all locked records
3475 4736 1 : JSB_REC0 NOVALUE =
3476 4737 1 ++
3477 4738 1 FUNCTIONAL DESCRIPTION:
3478 4739 1 Free all locked records. If successful or no records locked,
3479 4740 1 then return; otherwise, signal a fatal error.
3480 4741 1 FORMAL PARAMETERS:
3481 4742 1 NONE
3482 4743 1 IMPLICIT INPUTS:
3483 4744 1 NONE
3484 4745 1 IMPLICIT OUTPUTS:
3485 4746 1 RAB\$B_RAC record access field
3486 4747 1 ROUTINE VALUE:
3487 4748 1 NONE
3488 4749 1 SIDE EFFECTS:
3489 4750 1 -- SIGNALS any RMS errors
3490 4751 1
3491 4752 1 BEGIN
3492 4753 1 EXTERNAL REGISTER
3493 4754 1 CCB : REF BLOCK [, BYTE];
3494 4755 1 ++ Set the record access field in the RAB to sequential. Perform the FREE.
3495 4756 1 If RMS returns a failure status, signal the error.
3496 4757 1 --
3497 4758 1 CCB [RAB\$B_RAC] = RAB\$C_SEQ;
3498 4759 1 IF NOT \$FREE (RAB = .CCB)
3499 4760 1 THEN
3500 4761 1 IF .CCB [RAB\$L_STS] NEQ RM\$S_RNL
3501 4762 1 THEN
3502 4763 1 ++ An error was returned, check for "record not locked".
3503 4764 1 -- BASS\$STOP_IO (BASSK_IOERR_REC);
3504 4765 1 RETURN:
3505 4766 1 END; ! End of BASS\$REC_FEE
3506 4767 1
3507 4768 1
3508 4769 1
3509 4770 1
3510 4771 1
3511 4772 1
3512 4773 1
3513 4774 1
3514 4775 1
3515 4776 1
3516 4777 1
3517 4778 1
3518 4779 1
3519 4780 1
3520 4781 1
3521 4782 1
3522 4783 1
3523 4784 1
3524 4785 1
3525 4786 1
3526 4787 1
3527 4788 1

.EXTRN SYSSFREE

		1E	AB	94 00000 BASS\$REC_FEE::	
			5B	DD 00003	CLRB 30(CCB)
00000000G	00		01	FB 00005	PUSHL CCB
	14		50	E8 0000C	CALLS #1, SYSSFREE
000181A0	8F	08	AB	D1 0000F	BLBS R0, 1\$
			0A	13 00017	CMPL 8(CCB), #98720
00000000G	7E		01	CE 00019	BEQL 1\$
	00		01	FB 0001C	MNEG L #1. -(SP)
			05	00023 18:	CALLS #1, BASS\$STOP_10
					RSB

: 4775
: 4777
: 4780
: 4785
: 4788

; Routine Size: 36 bytes, Routine Base: _BASS\$CODE + 0C15

; 3528 4789 1

```
3530 ; 4790 1 GLOBAL ROUTINE BASS$REC_UPD (
3531 ; 4791 1     COUNT
3532 ; 4792 1     ) : JSB_DO_WRITE NOVALUE =
3533 ; 4793 1
3534 ; 4794 1
3535 ; 4795 1 ** FUNCTIONAL DESCRIPTION:
3536 ; 4796 1
3537 ; 4797 1     Update current record. If successful then return; otherwise, signal a fatal
3538 ; 4798 1     error.
3539 ; 4799 1
3540 ; 4800 1 FORMAL PARAMETERS:
3541 ; 4801 1     COUNT.rl.v          No. of bytes in record to update
3542 ; 4802 1
3543 ; 4803 1 IMPLICIT INPUTS:
3544 ; 4804 1     NONE
3545 ; 4805 1
3546 ; 4806 1 IMPLICIT OUTPUTS:
3547 ; 4807 1
3548 ; 4808 1     RABSB_RAC          record access field
3549 ; 4809 1     RABSW_RSZ           record size
3550 ; 4810 1
3551 ; 4811 1 ROUTINE VALUE:
3552 ; 4812 1     NONE
3553 ; 4813 1
3554 ; 4814 1 SIDE EFFECTS:
3555 ; 4815 1
3556 ; 4816 1     Update current record in file on this logical unit.
3557 ; 4817 1     SIGNALS any RMS errors
3558 ; 4818 1
3559 ; 4819 1
3560 ; 4820 1
3561 ; 4821 1 -- BEGIN
3562 ; 4822 1
3563 ; 4823 2
3564 ; 4824 2
3565 ; 4825 2 EXTERNAL REGISTER
3566 ; 4826 2     CCB : REF BLOCK [. BYTE];
3567 ; 4827 2
3568 ; 4828 2
3569 ; 4829 2 + Point RBF to the user buffer.
3570 ; 4830 2 | Set the record access field in the RAB to sequential. Perform the UPDATE.
3571 ; 4831 2 | If RMS returns a failure status, signal the error.
3572 ; 4832 2 |
3573 ; 4833 2 | CCB [RAB$L_RBF] = .CCB [RAB$L_UBF];
3574 ; 4834 2 | CCB [RABSW_RSZ] = .COUNT;
3575 ; 4835 2 | CCB [RABSB_RAC] = RAB$C_SEQ;
3576 ; 4836 2
3577 ; 4837 2 | IF NOT SUPDATE (RAB = .CCB) THEN BASS$STOP_IO (BASS$K_ICERR_REC);
3578 ; 4838 2
3579 ; 4839 2 | Point LUBSA_RBUF_ADR to the buffer used by RMS for MOVE.
3580 ; 4840 2 | CCB [LUBSA_RBUF_ADR] = .CCB [RAB$L_UBF];
3581 ; 4841 2 | RETURN;
3582 ; 4842 2 | END;
3583 ; 4843 2
3584 ; 4844 2     ! End of BASS$REC_UPD
```

.EXTRN SYSSUPDATE

28 AB	24 AB DD 00000 BASS\$REC_UPD::	MOVL 36(CC(B) 40(CC(B))	4833
22 AB	1E 50 B0 00005	MOVW COUNT 34(CC(B))	4834
	AB 94 00009	CLRB 30(CC(B))	4835
00000000G 00	58 DD 0000C	PUSHL CCB	4837
0A	01 FB 0000E	CALLS #1. SYSSUPDATE	
7E	50 EB 00015	BLBS R0, 1S	
00000000G 00	01 CE 00018	MNEGL #1. -(SP)	
EC AB	24 AB DD 00022 1\$: 05 00027	CALLS #1. BASS\$STOP_10	4842
		MOVL 36(CC(B), -20(CC(B))	4844
		RSB	

: Routine Size: 40 bytes. Routine Base: _BASS\$CODE + 0C39

: 3585 4845 1

```

3587 4846 1 GLOBAL ROUTINE BASS$REC_RSE           ! RESTORE (sequential) to beginning of file
3588 4847 1 : JSB_REC0 NOVALUE =
3589 4848 1 ++
3590 4849 1 FUNCTIONAL DESCRIPTION:
3591 4850 1 Rewind the file. If successful then return; otherwise, signal a fatal
3592 4851 1 error.
3593 4852 1 FORMAL PARAMETERS:
3594 4853 1
3595 4854 1
3596 4855 1
3597 4856 1
3598 4857 1
3599 4858 1
3600 4859 1
3601 4860 1
3602 4861 1
3603 4862 1
3604 4863 1
3605 4864 1
3606 4865 1 RABSB_RAC      record access field
3607 4866 1
3608 4867 1 ROUTINE VALUE:
3609 4868 1
3610 4869 1
3611 4870 1
3612 4871 1 SIDE EFFECTS:
3613 4872 1
3614 4873 1
3615 4874 1
3616 4875 1
3617 4876 2 BEGIN
3618 4877 2
3619 4878 2 EXTERNAL REGISTER
3620 4879 2   CCB : REF BLOCK [, BYTE];
3621 4880 2
3622 4881 2
3623 4882 2
3624 4883 2
3625 4884 2
3626 4885 2
3627 4886 2
3628 4887 2
3629 4888 2
3630 4889 2
3631 4890 2
3632 4891 1 RETURN;          ! End of BASS$REC_RSE
END;
```

.EXTRN SYSSREWIND

	1E AB 94 00000 BASS\$REC_RSE::	
	CLRB	30(CCB)
0000000G 00	5B DD 00003	PUSHL CCB
	01 FB 00005	CALLS #1. SYSSREWIND
0A	50 E8 0000C	BLBS R0: 18
7E	01 CE 0000F	MNEG L #1, -(SP)

4886
4888

BASS\$REC_PROC
1-095

B 12
16-Sep-1984 01:01:12 VAX-11 Bliss-32 v4.0-742
14-Sep-1984 11:56:35 [BASRTL.SRC]BASRECPRO.B32;1

Page 100
(38)

00000000G 00 01 FB 00012 CALLS #1, BASS\$STOP_10
 05 00019 18: RSB

; 4891

: Routine Size: 26 bytes, Routine Base: _BASS\$CODE + 0C61

: 3633 4892 1

```

: 3635      4893 1 GLOBAL ROUTINE BASS$REC_RIN (
: 3636          4894 1   KEY_NO) : JSB_REC_IND NOVALUE =
: 3637          4895 1
: 3638          4896 1
: 3639          4897 1
: 3640          4898 1
: 3641          4899 1
: 3642          4900 1
: 3643          4901 1
: 3644          4902 1
: 3645          4903 1
: 3646          4904 1
: 3647          4905 1
: 3648          4906 1
: 3649          4907 1
: 3650          4908 1
: 3651          4909 1
: 3652          4910 1
: 3653          4911 1
: 3654          4912 1
: 3655          4913 1
: 3656          4914 1
: 3657          4915 1
: 3658          4916 1
: 3659          4917 1
: 3660          4918 1
: 3661          4919 1
: 3662          4920 1
: 3663          4921 1
: 3664          4922 1
: 3665          4923 1
: 3666          4924 2
: 3667          4925 2
: 3668          4926 2
: 3669          4927 2
: 3670          4928 2
: 3671          4929 2
: 3672          4930 2
: 3673          4931 2
: 3674          4932 2
: 3675          4933 2
: 3676          4934 2
: 3677          4935 2
: 3678          4936 2
: 3679          4937 2
: 3680          4938 2
: 3681          4939 2
: 3682          4940 2
: 3683          4941 1

        ++ FUNCTIONAL DESCRIPTION:
        Rewind the file. If successful then return; otherwise, signal a fatal error.

        FORMAL PARAMETERS:
        KEY_NO.rl.v           key of reference

        IMPLICIT INPUTS:
        NONE

        IMPLICIT OUTPUTS:
        RABSB_KRF             key of reference
        RABSB_RAC              record access field

        ROUTINE VALUE:
        NONE

        SIDE EFFECTS:
        SIGNALS any RMS errors
-- BEGIN

        EXTERNAL REGISTER
        CCB : REF BLOCK [, BYTE];

        /* Set the key of reference.
        Set the record access field in the RAB to key. Perform the REWIND.
        If RMS returns a failure status, signal the error.

        CCB [RABSB_KRF] = .KEY_NO;
        CCB [RABSB_RAC] = RABSC_KEY;
        IF NOT $REWIND (RAB = .CCB) THEN BASS$STOP_IO (BASSK_IOERR_REC);
        RETURN;
        END;

        ! End of BASS$REC_RIN

```

35 AB	50 90 00000 BASS\$REC_RIN::	
1E AB	01 90 00004	MOV B KEY_NO, 53(CCB)
		MOV B #1,-30(CCB)

: 4935
: 4936

00000000G	00	5B DD 00008	PUSHL	CCB
	0A	01 FB 0000A	CALLS	#1, SYSSREWIND
	7E	50 E8 00011	BLBS	R0, 1\$
00000000G	00	01 CE 00014	MNEGL	#1, -(SP)
		01 FB 00017	CALLS	#1, BASS\$STOP_10
		05 0001E 1\$:	RSB	

: 4938

: 4941

; Routine Size: 31 bytes. Routine Base: _BASS\$CODE + 0C7B

; 3684 4942 1

```

3686      6943 1 GLOBAL ROUTINE BASS$REC_SSE           ! SCRATCH (sequential) a record
3687      6944 1 : JSB_REC0 NOVALUE =
3688      6945 1
3689      6946 1 ++
3690      6947 1 FUNCTIONAL DESCRIPTION:
3691      6948 1 Truncate this file. If successful then return; otherwise, signal a fatal
3692      6949 1 error.
3693      6950 1
3694      6951 1 FORMAL PARAMETERS:
3695      6952 1
3696      6953 1
3697      6954 1 NONE
3698      6955 1
3699      6956 1 IMPLICIT INPUTS:
3700      6957 1
3701      6958 1 NONE
3702      6959 1
3703      6960 1 IMPLICIT OUTPUTS:
3704      6961 1
3705      6962 1 RAB$B_RAC          record access field
3706      6963 1
3707      6964 1 ROUTINE VALUE:
3708      6965 1
3709      6966 1 NONE
3710      6967 1
3711      6968 1 SIDE EFFECTS:
3712      6969 1
3713      6970 1
3714      6971 1
3715      6972 1
3716      6973 2 BEGIN
3717      6974 2
3718      6975 2 EXTERNAL REGISTER
3719      6976 2   CCB : REF BLOCK [, BYTE];
3720      6977 2
3721      6978 2
3722      6979 2   !+ Set the record access field in the RAB to sequential. Perform the TRUNCATE.
3723      6980 2   !+ If RMS returns a failure status, signal the error.
3724      6981 2   !-
3725      6982 2
3726      6983 2   CCB [RAB$B_RAC] = RAB$C_SEQ;
3727      6984 2
3728      6985 2   IF NOT STRUNCATE (RAB = .(CCB) THEN BASS$STOP_IO (BASS$IOERR_REC);
3729      6986 2
3730      6987 2   RETURN;
3731      6988 1   END:           ! End of BASS$REC_SSE

```

.EXTRN SYS\$TRUNCATE

	1E AB 94 00000 BASS\$REC_SSE::	
0000000G 00	5B DD 00003	CLRB 30(CCB)
0A	01 FB 00005	PUSHL CCB
7E	50 EB 0000C	CALLS #1, SYS\$TRUNCATE
	01 CE 0000F	BLBS R0, 1\$
		MNEGL #1, -(SP)

: 4983
: 4985

BASS\$REC_PROC
1-095

F 12
16-Sep-1984 01:01:12 VAX-11 Bliss-32 V4.0-742
14-Sep-1984 11:56:35 [BASRTL.SRC]BASREC[PRO.B32;1]

Page 104
(40)

00000000G 00 01 FB 00012 CALLS #1, BASS\$STOP_10
05 00019 1\$: RSB

; 4988

; Routine Size: 26 bytes, Routine Base: _BASSCODE + '0C9A

; 3732 4989 1

```

3734 4990 1 ROUTINE PUT_ERROR (
3735 4991 1 SIGNAL OR STOP
3736 4992 1 ) : CALL_CCB NOVALUE =
3737 4993 1
3738 4994 1 ++
3739 4995 1 FUNCTIONAL DESCRIPTION:
3740 4996 1
3741 4997 1 Here on SPUT errors, check for Record stream active error (RMSS_RSA)
3742 4998 1 If this error, WAIT until not active and try SPUT again.
3743 4999 1 This recovers from AST I/O which can occur out of the middle
3744 5000 1 of synchronous I/O at non-AST level.
3745 5001 1
3746 5002 1 CALLING SEQUENCE:
3747 5003 1
3748 5004 1 PUT_ERROR (signal_or_stop)
3749 5005 1
3750 5006 1 FORMAL PARAMETERS:
3751 5007 1
3752 5008 1 SIGNAL_OR_STOP.rl.v whether to signal or stop
3753 5009 1
3754 5010 1 IMPLICIT INPUTS:
3755 5011 1
3756 5012 1 CCB Adr. of current LUB/ISB/RAB
3757 5013 1
3758 5014 1 IMPLICIT OUTPUTS:
3759 5015 1
3760 5016 1 LUBSV_OUTBUF_DR Cleared to indicate clean buffer
3761 5017 1
3762 5018 1 ROUTINE VALUE:
3763 5019 1
3764 5020 1
3765 5021 1
3766 5022 1
3767 5023 1
3768 5024 1
3769 5025 1 SIDE EFFECTS:
3770 5026 1 SWAITs and then tries SPUT again, until success or any error
3771 5027 1 except record stream active.
3772 5028 2
3773 5029 2
3774 5030 2
3775 5031 2
3776 5032 2
3777 5033 2
3778 5034 2
3779 5035 2
3780 5036 2
3781 5037 2
3782 5038 2
3783 5039 2
3784 5040 2
3785 5041 2
3786 5042 2
3787 5043 2
3788 5044 2
3789 5045 2
3790 5046 2
3791 5047 2
3792 5048 2
3793 5049 2
3794 5050 2
3795 5051 2
3796 5052 2
3797 5053 2
3798 5054 2
3799 5055 2
3800 5056 2
3801 5057 2
3802 5058 2
3803 5059 2
3804 5060 2
3805 5061 2
3806 5062 2
3807 5063 2
3808 5064 2
3809 5065 2
3810 5066 2
3811 5067 2
3812 5068 2
3813 5069 2
3814 5070 2
3815 5071 2
3816 5072 2
3817 5073 2
3818 5074 2
3819 5075 2
3820 5076 2
3821 5077 2
3822 5078 2
3823 5079 2
3824 5080 2
3825 5081 2
3826 5082 2
3827 5083 2
3828 5084 2
3829 5085 2
3830 5086 2
3831 5087 2
3832 5088 2
3833 5089 2
3834 5090 2
3835 5091 2
3836 5092 2
3837 5093 2
3838 5094 2
3839 5095 2
3840 5096 2
3841 5097 2
3842 5098 2
3843 5099 2
3844 5100 2
3845 5101 2
3846 5102 2
3847 5103 2
3848 5104 2
3849 5105 2
3850 5106 2
3851 5107 2
3852 5108 2
3853 5109 2
3854 5110 2
3855 5111 2
3856 5112 2
3857 5113 2
3858 5114 2
3859 5115 2
3860 5116 2
3861 5117 2
3862 5118 2
3863 5119 2
3864 5120 2
3865 5121 2
3866 5122 2
3867 5123 2
3868 5124 2
3869 5125 2
3870 5126 2
3871 5127 2
3872 5128 2
3873 5129 2
3874 5130 2
3875 5131 2
3876 5132 2
3877 5133 2
3878 5134 2
3879 5135 2
3880 5136 2
3881 5137 2
3882 5138 2
3883 5139 2
3884 5140 2
3885 5141 2
3886 5142 2
3887 5143 2
3888 5144 2
3889 5145 2
3890 5146 2
3891 5147 2
3892 5148 2
3893 5149 2
3894 5150 2
3895 5151 2
3896 5152 2
3897 5153 2
3898 5154 2
3899 5155 2
3900 5156 2
3901 5157 2
3902 5158 2
3903 5159 2
3904 5160 2
3905 5161 2
3906 5162 2
3907 5163 2
3908 5164 2
3909 5165 2
3910 5166 2
3911 5167 2
3912 5168 2
3913 5169 2
3914 5170 2
3915 5171 2
3916 5172 2
3917 5173 2
3918 5174 2
3919 5175 2
3920 5176 2
3921 5177 2
3922 5178 2
3923 5179 2
3924 5180 2
3925 5181 2
3926 5182 2
3927 5183 2
3928 5184 2
3929 5185 2
3930 5186 2
3931 5187 2
3932 5188 2
3933 5189 2
3934 5190 2
3935 5191 2
3936 5192 2
3937 5193 2
3938 5194 2
3939 5195 2
3940 5196 2
3941 5197 2
3942 5198 2
3943 5199 2
3944 5200 2
3945 5201 2
3946 5202 2
3947 5203 2
3948 5204 2
3949 5205 2
3950 5206 2
3951 5207 2
3952 5208 2
3953 5209 2
3954 5210 2
3955 5211 2
3956 5212 2
3957 5213 2
3958 5214 2
3959 5215 2
3960 5216 2
3961 5217 2
3962 5218 2
3963 5219 2
3964 5220 2
3965 5221 2
3966 5222 2
3967 5223 2
3968 5224 2
3969 5225 2
3970 5226 2
3971 5227 2
3972 5228 2
3973 5229 2
3974 5230 2
3975 5231 2
3976 5232 2
3977 5233 2
3978 5234 2
3979 5235 2
3980 5236 2
3981 5237 2
3982 5238 2
3983 5239 2
3984 5240 2
3985 5241 2
3986 5242 2
3987 5243 2
3988 5244 2
3989 5245 2
3990 5246 2
3991 5247 2
3992 5248 2
3993 5249 2
3994 5250 2
3995 5251 2
3996 5252 2
3997 5253 2
3998 5254 2
3999 5255 2
4000 5256 2
4001 5257 2
4002 5258 2
4003 5259 2
4004 5260 2
4005 5261 2
4006 5262 2
4007 5263 2
4008 5264 2
4009 5265 2
4010 5266 2
4011 5267 2
4012 5268 2
4013 5269 2
4014 5270 2
4015 5271 2
4016 5272 2
4017 5273 2
4018 5274 2
4019 5275 2
4020 5276 2
4021 5277 2
4022 5278 2
4023 5279 2
4024 5280 2
4025 5281 2
4026 5282 2
4027 5283 2
4028 5284 2
4029 5285 2
4030 5286 2
4031 5287 2
4032 5288 2
4033 5289 2
4034 5290 2
4035 5291 2
4036 5292 2
4037 5293 2
4038 5294 2
4039 5295 2
4040 5296 2
4041 5297 2
4042 5298 2
4043 5299 2
4044 5300 2
4045 5301 2
4046 5302 2
4047 5303 2
4048 5304 2
4049 5305 2
4050 5306 2
4051 5307 2
4052 5308 2
4053 5309 2
4054 5310 2
4055 5311 2
4056 5312 2
4057 5313 2
4058 5314 2
4059 5315 2
4060 5316 2
4061 5317 2
4062 5318 2
4063 5319 2
4064 5320 2
4065 5321 2
4066 5322 2
4067 5323 2
4068 5324 2
4069 5325 2
4070 5326 2
4071 5327 2
4072 5328 2
4073 5329 2
4074 5330 2
4075 5331 2
4076 5332 2
4077 5333 2
4078 5334 2
4079 5335 2
4080 5336 2
4081 5337 2
4082 5338 2
4083 5339 2
4084 5340 2
4085 5341 2
4086 5342 2
4087 5343 2
4088 5344 2
4089 5345 2
4090 5346 2
4091 5347 2
4092 5348 2
4093 5349 2
4094 5350 2
4095 5351 2
4096 5352 2
4097 5353 2
4098 5354 2
4099 5355 2
4100 5356 2
4101 5357 2
4102 5358 2
4103 5359 2
4104 5360 2
4105 5361 2
4106 5362 2
4107 5363 2
4108 5364 2
4109 5365 2
4110 5366 2
4111 5367 2
4112 5368 2
4113 5369 2
4114 5370 2
4115 5371 2
4116 5372 2
4117 5373 2
4118 5374 2
4119 5375 2
4120 5376 2
4121 5377 2
4122 5378 2
4123 5379 2
4124 5380 2
4125 5381 2
4126 5382 2
4127 5383 2
4128 5384 2
4129 5385 2
4130 5386 2
4131 5387 2
4132 5388 2
4133 5389 2
4134 5390 2
4135 5391 2
4136 5392 2
4137 5393 2
4138 5394 2
4139 5395 2
4140 5396 2
4141 5397 2
4142 5398 2
4143 5399 2
4144 5400 2
4145 5401 2
4146 5402 2
4147 5403 2
4148 5404 2
4149 5405 2
4150 5406 2
4151 5407 2
4152 5408 2
4153 5409 2
4154 5410 2
4155 5411 2
4156 5412 2
4157 5413 2
4158 5414 2
4159 5415 2
4160 5416 2
4161 5417 2
4162 5418 2
4163 5419 2
4164 5420 2
4165 5421 2
4166 5422 2
4167 5423 2
4168 5424 2
4169 5425 2
4170 5426 2
4171 5427 2
4172 5428 2
4173 5429 2
4174 5430 2
4175 5431 2
4176 5432 2
4177 5433 2
4178 5434 2
4179 5435 2
4180 5436 2
4181 5437 2
4182 5438 2
4183 5439 2
4184 5440 2
4185 5441 2
4186 5442 2
4187 5443 2
4188 5444 2
4189 5445 2
4190 5446 2
4191 5447 2
4192 5448 2
4193 5449 2
4194 5450 2
4195 5451 2
4196 5452 2
4197 5453 2
4198 5454 2
4199 5455 2
4200 5456 2
4201 5457 2
4202 5458 2
4203 5459 2
4204 5460 2
4205 5461 2
4206 5462 2
4207 5463 2
4208 5464 2
4209 5465 2
4210 5466 2
4211 5467 2
4212 5468 2
4213 5469 2
4214 5470 2
4215 5471 2
4216 5472 2
4217 5473 2
4218 5474 2
4219 5475 2
4220 5476 2
4221 5477 2
4222 5478 2
4223 5479 2
4224 5480 2
4225 5481 2
4226 5482 2
4227 5483 2
4228 5484 2
4229 5485 2
4230 5486 2
4231 5487 2
4232 5488 2
4233 5489 2
4234 5490 2
4235 5491 2
4236 5492 2
4237 5493 2
4238 5494 2
4239 5495 2
4240 5496 2
4241 5497 2
4242 5498 2
4243 5499 2
4244 5500 2
4245 5501 2
4246 5502 2
4247 5503 2
4248 5504 2
4249 5505 2
4250 5506 2
4251 5507 2
4252 5508 2
4253 5509 2
4254 5510 2
4255 5511 2
4256 5512 2
4257 5513 2
4258 5514 2
4259 5515 2
4260 5516 2
4261 5517 2
4262 5518 2
4263 5519 2
4264 5520 2
4265 5521 2
4266 5522 2
4267 5523 2
4268 5524 2
4269 5525 2
4270 5526 2
4271 5527 2
4272 5528 2
4273 5529 2
4274 5530 2
4275 5531 2
4276 5532 2
4277 5533 2
4278 5534 2
4279 5535 2
4280 5536 2
4281 5537 2
4282 5538 2
4283 5539 2
4284 5540 2
4285 5541 2
4286 5542 2
4287 5543 2
4288 5544 2
4289 5545 2
4290 5546 2
4291 5547 2
4292 5548 2
4293 5549 2
4294 5550 2
4295 5551 2
4296 5552 2
4297 5553 2
4298 5554 2
4299 5555 2
4300 5556 2
4301 5557 2
4302 5558 2
4303 5559 2
4304 5560 2
4305 5561 2
4306 5562 2
4307 5563 2
4308 5564 2
4309 5565 2
4310 5566 2
4311 5567 2
4312 5568 2
4313 5569 2
4314 5570 2
4315 5571 2
4316 5572 2
4317 5573 2
4318 5574 2
4319 5575 2
4320 5576 2
4321 5577 2
4322 5578 2
4323 5579 2
4324 5580 2
4325 5581 2
4326 5582 2
4327 5583 2
4328 5584 2
4329 5585 2
4330 5586 2
4331 5587 2
4332 5588 2
4333 5589 2
4334 5590 2
4335 5591 2
4336 5592 2
4337 5593 2
4338 5594 2
4339 5595 2
4340 5596 2
4341 5597 2
4342 5598 2
4343 5599 2
4344 5600 2
4345 5601 2
4346 5602 2
4347 5603 2
4348 5604 2
4349 5605 2
4350 5606 2
4351 5607 2
4352 5608 2
4353 5609 2
4354 5610 2
4355 5611 2
4356 5612 2
4357 5613 2
4358 5614 2
4359 5615 2
4360 5616 2
4361 5617 2
4362 5618 2
4363 5619 2
4364 5620 2
4365 5621 2
4366 5622 2
4367 5623 2
4368 5624 2
4369 5625 2
4370 5626 2
4371 5627 2
4372 5628 2
4373 5629 2
4374 5630 2
4375 5631 2
4376 5632 2
4377 5633 2
4378 5634 2
4379 5635 2
4380 5636 2
4381 5637 2
4382 5638 2
4383 5639 2
4384 5640 2
4385 5641 2
4386 5642 2
4387 5643 2
4388 5644 2
4389 5645 2
4390 5646 2
4391 5647 2
4392 5648 2
4393 5649 2
4394 5650 2
4395 5651 2
4396 5652 2
4397 5653 2
4398 5654 2
4399 5655 2
4400 5656 2
4401 5657 2
4402 5658 2
4403 5659 2
4404 5660 2
4405 5661 2
4406 5662 2
4407 5663 2
4408 5664 2
4409 5665 2
4410 5666 2
4411 5667 2
4412 5668 2
4413 5669 2
4414 5670 2
4415 5671 2
4416 5672 2
4417 5673 2
4418 5674 2
4419 5675 2
4420 5676 2
4421 5677 2
4422 5678 2
4423 5679 2
4424 5680 2
4425 5681 2
4426 5682 2
4427 5683 2
4428 5684 2
4429 5685 2
4430 5686 2
4431 5687 2
4432 5688 2
4433 5689 2
4434 5690 2
4435 5691 2
4436 5692 2
4437 5693 2
4438 5694 2
4439 5695 2
4440 5696 2
4441 5697 2
4442 5698 2
4443 5699 2
4444 5700 2
4445 5701 2
4446 5702 2
4447 5703 2
4448 5704 2
4449 5705 2
4450 5706 2
4451 5707 2
4452 5708 2
4453 5709 2
4454 5710 2
4455 5711 2
4456 5712 2
4457 5713 2
4458 5714 2
4459 5715 2
4460 5716 2
4461 5717 2
4462 5718 2
4463 5719 2
4464 5720 2
4465 5721 2
4466 5722 2
4467 5723 2
4468 5724 2
4469 5725 2
4470 5726 2
4471 5727 2
4472 5728 2
4473 5729 2
4474 5730 2
4475 5731 2
4476 5732 2
4477 5733 2
4478 5734 2
4479 5735 2
4480 5736 2
4481 5737 2
4482 5738 2
4483 5739 2
4484 5740 2
4485 5741 2
4486 5742 2
4487 5743 2
4488 5744 2
4489 5745 2
4490 5746 2
4491 5747 2
4492 5748 2
4493 5749 2
4494 5750 2
4495 5751 2
4496 5752 2
4497 5753 2
4498 5754 2
4499 5755 2
4500 5756 2
4501 5757 2
4502 5758 2
4503 5759 2
4504 5760 2
4505 5761 2
4506 5762 2
4507 5763 2
4508 5764 2
4509 5765 2
4510 5766 2
4511 5767 2
4512 5768 2
4513 5769 2
4514 5770 2
4515 5771 2
4516 5772 2
4517 5773 2
4518 5774 2
4519 5775 2
4520 5776 2
4521 5777 2
4522 5778 2
4523 5779 2
4524 5780 2
4525 5781 2
4526 5782 2
4527 5783 2
4528 5784 2
4529 5785 2
4530 5786 2
4531 5787 2
4532 5788 2
4533 5789 2
4534 5790 2
4535 5791 2
4536 5792 2
4537 5793 2
4538 5794 2
4539 5795 2
4540 5796 2
4541 5797 2
4542 5798 2
4543 5799 2
4544 5800 2
4545 5801 2
4546 5802 2
4547 5803 2
4548 5804 2
4549 5805 2
4550 5806 2
4551 5807 2
4552 5808 2
4553 5809 2
4554 5810 2
4555 5811 2
4556 5812 2
4557 5813 2
4558 5814 2
4559 5815 2
4560 5816 2
4561 5817 2
4562 5818 2
4563 5819 2
4564 5820 2
4565 5821 2
4566 5822 2
4567 5823 2
4568 5824 2
4569 5825 2
4570 5826 2
4571 5827 2
4572 5828 2
4573 5829 2
4574 5830 2
4575 5831 2
4576 5832 2
4577 5833 2
4578 5834 2
4579 5835 2
4580 5836 2
4581 5837 2
4582 5838 2
4583 5839 2
4584 5840 2
4585 5841 2
4586 5842 2
4587 5843 2
4588 5844 2
4589 5845 2
4590 5846 2
4591 5847 2
4592 5848 2
4593 5849 2
4594 5850 2
4595 5851 2
4596 5852 2
4597 5853 2
4598 5854 2
4599 5855 2
4600 5856 2
4601 5857 2
4602 5858 2
4603 5859 2
4604 5860 2
4605 5861 2
4606 5862 2
4607 5863 2
4608 5864 2
4609 5865 2
4610 5866 2
4611 5867 2
4612 5868 2
4613 5869 2
4614 5870 2
4615 5871 2
4616 5872 2
4617 5873 2
4618 5874 2
4619 5875 2
4620 5876 2
4621 5877 2
4622 5878 2
4623 5879 2
4624 5880 2
4625 5881 2
4626 5882 2
4627 5883 2
4628 5884 2
4629 5885 2
4630 5886 2
4631 5887 2
4632 5888 2
4633 5889 2
4634 5890 2
4635 5891 2
4636 5892 2
4637 5893 2
```

```

: 3791      5047 3
: 3792      5048 3
: 3793      5049 3
: 3794      5050 3
: 3795      5051 3
: 3796      5052 3
: 3797      5053 3
: 3798      5054 2
: 3799      5055 2
: 3800      5056 1

      IF .SIGNAL_OR_STOP EQL K_SIGNAL
      THEN BAS$SIGNAL_IO (BASSK_IOERR_REC)
      ELSE BAS$$STOP_IO (BASSK_IOERR_REC);
      END;
      RETURN;
      END;

```

! End of PUT_ERROR

0000 00000 PUT_ERROR:							
							.WORD
000182DA	BF	08	AB	D1	00002	1\$:	Cmpl 8(CC(B), #99034
			14	12	0000A		BNEQ 2\$
00000000G	00		5B	DD	0000C		PUSHL CCB
00000000G	00		01	FB	0000E		CALLS #1, SYSSWAIT
			5B	DD	00015		PUSHL CCB
			01	FB	00017		CALLS #1, SYSSPUT
			E2	11	0001E		BRB 1\$
			FE	1F	00020	2\$:	BLBS 8(CC(B), 4\$
			AB	08	8A	00024	BIC(B2) #8, -2(CC(B)
			01	04	AC	D1 00028	Cmpl SIGNAL_OR_STOP, #1
					0B	12 0002C	BNEQ 3\$
00000000G	00				01	CE 0002E	MNEG L #1, -(SP)
					01	FB 00031	CALLS #1, BAS\$SIGNAL_IO
					04	00038	RET
00000000G	00				01	CE 00039	3\$: MNEG L #1, -(SP)
					01	FB 0003C	CALLS #1, BAS\$\$STOP_IO
					04	00043	4\$: RET

: Routine Size: 68 bytes, Routine Base: _BASSCODE + 0CB4

3802 5057 1 ROUTINE GET_ERROR (! Here on error on \$GET
3803 5058 1 SIGNAL OR STOP ! parameter to signal(continue) or stop
3804 5059 1) : CALL_CCB NOVALUE =
3805 5060 1
3806 5061 1 ++
3807 5062 1 FUNCTIONAL DESCRIPTION:
3808 5063 1
3809 5064 1 Here on \$GET errors, check for Record stream active error (RMSS_RSA)
3810 5065 1 If this error, WAIT until not active and try \$GET again.
3811 5066 1 This recovers from AST I/O which can occur out of the middle
3812 5067 1 of synchronous I/O at non-AST level.
3813 5068 1
3814 5069 1 CALLING SEQUENCE:
3815 5070 1 JSB GET_ERROR ()
3816 5071 1
3817 5072 1 FORMAL PARAMETERS:
3818 5073 1
3819 5074 1
3820 5075 1
3821 5076 1
3822 5077 1
3823 5078 1
3824 5079 1
3825 5080 1 CCB Adr. of current LUB/ISB/RAB
3826 5081 1
3827 5082 1
3828 5083 1
3829 5084 1
3830 5085 1
3831 5086 1
3832 5087 1
3833 5088 1
3834 5089 1
3835 5090 1
3836 5091 1
3837 5092 1
3838 5093 1
3839 5094 1
3840 5095 2
3841 5096 2
3842 5097 2
3843 5098 2 EXTERNAL REGISTER
3844 5099 2 CCB : REF BLOCK [. BYTE];
3845 5100 2
3846 5101 2
3847 5102 2 Set the prompt buffer length to zero so that error followed by RESUME will not
3848 5103 2 keep concatenating the prompt buffer.
3849 5104 2
3850 5105 2
3851 5106 2
3852 5107 2
3853 5108 2
3854 5109 2
3855 5110 2
3856 5111 2
3857 5112 2 IF .CCB [ISBSB_STM_TYPE] EQL ISBK_ST_TY_INL AND .CCB [RABSW_RSZ] EQLU 1 AND .(.CCB [RABSL_RBF])<0, 8>
3858 5113 2 EQLU BASSK_CONTROL_Z
THEN RETURN;

```

3859      5114 2 WHILE .CCB [RABSL_STS] EQL RMSS_RSA DO
3860      5115 3 BEGIN
3861      5116 3 SWAIT (RAB = .CCB);
3862      5117 3 SGET (RAB = .CCB)
3863      5118 2 END;
3864      5119 2
3865      5120 2 IF NOT .CCB [RABSL_STS]
3866      5121 2 THEN
3867      5122 2
3868      5123 2 !+ Check the input parameter to see if we should signal or stop.
3869      5124 2 !-
3870      5125 2
3871      5126 2
3872      5127 2 IF .SIGNAL_OR_STOP EQL K_SIGNAL
3873      5128 2 THEN BASS$SIGNAL_IO (BASS$K_IOERR_REC)
3874      5129 2 ELSE BASS$STOP_IO (BASS$K_IOERR_REC);
3875      5130 2
3876      5131 2
3877      5132 2
3878      5133 2 RETURN;
3879      5134 1 END;                                ! End of GET_ERROR

```

0000 00000 GET_ERROR:						
				.WORD	Save nothing	
20	FF71	34	AB 94 00002	CLRB	52(CCB)	
			CB 91 00005	CMPB	-143(CCB), #32	
01		22	OC 12 0000A	BNEQ	1\$	
			AB B1 0000C	CMPW	34(CCB), #1	
		06	12 00010	BNEQ	1\$	
1A		28	BB 91 00012	CMPB	@40(CCB), #26	
			3D 13 00016	BEQL	4\$	
000182DA	8F	08	AB D1 00018 1\$:	CMPL	8(CCB), #99034	
			14 12 00020	BNEQ	2\$	
			5B DD 00022	PUSHL	CCB	
00000000G	00		01 FB 00024	CALLS	#1, SY\$SWAIT	
00000000G	00		5B DD 0002B	PUSHL	CCB	
			01 FB 0002D	CALLS	#1, SY\$GET	
		E2	11 00034	BRB	1\$	
		18	AB E8 00036 2\$:	BLBS	8(CCB), 4\$	
		01	04 AC D1 0003A	CMPL	SIGNAL_OR_STOP, #1	
			0B 12 0003E	BNEQ	3\$	
00000000G	00	7E	01 CE 00040	MNEG	#1, -(SP)	
			01 FB 00043	CALLS	#1, BASS\$SIGNAL_IO	
			04 0004A	RET		
00000000G	00	7E	01 CE 0004B 3\$:	MNEG	#1, -(SP)	
			01 FB 0004E	CALLS	#1, BASS\$STOP_IO	
			04 00055 4\$:	RET		

; Routine Size: 86 bytes. Routine Base: _BASS\$CODE + 0CF8

: 3880 5135 1 END
: 3881 5136 1

BASS\$REC_PROC
1-095

: 3882 5137 0 ELUDOM

K 12
16-Sep-1984 01:01:12
14-Sep-1984 11:56:35

VAX-11 Bliss-32 v4.0-742
[BASRTL.SRC]BASRECPRO.B32;1

Page 109
(42)

BASS\$REC_WF9== BASS\$REC_WSL9
BASS\$REC_WF1== BASS\$REC_WSL1
BASS\$REC_WF0== BASS\$REC_WSL0

PSECT SUMMARY

Name	Bytes	Attributes
-BASS\$DATA	6 NOVEC, WRT, RD ,NOEXE,NOSHR, LCL, REL, CON, PIC,ALIGN(2)	
-BASS\$CODE	3406 NOVEC,NOWRT, RD , EXE, SHR, LCL, REL, CON, PIC,ALIGN(2)	

Library Statistics

File	Total	Symbols	Loaded	Percent	Pages Mapped	Processing Time
\$_\$255\$DUA28:[SYSLIB]STARLET.L32;1	9776	44	0	0	581	00:01.2

COMMAND QUALIFIERS

BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACE/LIS=LISS:BASRECPRO/OBJ=OBJ\$:BASRECPRO MSRC\$:BASRECPRO/UPDATE=(ENH\$:BASRECPRO)

: Size: 3397 code + 15 data bytes
: Run Time: 01:16.8
: Elapsed Time: 02:50.9
: Lines/CPU Min: 4014
: Lexemes/CPU-Min: 26026
: Memory Used: 249 pages
: Compilation complete

0030 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

BASRAD50
LTS

BASPUT
LTS

BASRECPRO
LTS

BASRANDOM
LTS

BASRESTAR
LTS

BASREMAP
LTS

BASRESTOR
LTS

BASRIGHT
LTS

BASRSTSU
LTS